Unsupervised Conflict-Free Ontology Evolution Without Removing Axioms

Thomas Scharrenbach[1], Claudia d'Amato[2], Nicola Fanizzi[2], Rolf Grütter[1], Bettina Waldvogel[1], and Abraham Bernstein[3]

[1] Swiss Federal Institute for Forest, Snow and Landscape Research WSL Birmensdorf, Switzerland
{thomas.scharrenbach, rolf.gruetter, bettina.waldvogel}@wsl.ch
[2] Università degli Studi di Bari Bari, Italy {claudia.damato, fanizzi}@di.uniba.it
[3] University of Zurich, Department of Informatics Zurich, Switzerland
{bernstein}@ifi.uzh.ch

Abstract. In the beginning of the Semantic Web, ontologies were usually constructed once by a single knowledge engineer and then used as a static conceptualization of some domain. Nowadays, knowledge bases are increasingly dynamically evolving and incorporate new knowledge from different heterogeneous domains – some of which is even contributed by casual users (i.e., non-knowledge engineers) or even software agents. Given that ontologies are based on the rather strict formalism of Description Logics and their inference procedures, conflicts are likely to occur during ontology evolution. Conflicts, in turn, may cause an ontological knowledge base to become inconsistent and making reasoning impossible. Hence, every formalism for ontology evolution should provide a mechanism for resolving conflicts.
In this paper we provide a general framework for conflict-free ontology evolution without changing the knowledge representation. Using a variant of Lehmann's Default Logics and Probabilistic Description Logics, we can invalidate unwanted implicit inferences without removing explicitly stated axioms. We show that this method outperforms classical ontology repair w.r.t. the amount of information lost while allowing for automatic conflict-solving when evolving ontologies.

1 Introduction

Knowledge in the Semantic Web is represented by ontologies expressed in the Web Ontology Language OWL. The current standard, OWL2 [1], defines different profiles all of which have some Description Logics as a rough syntactic variant, allowing for different levels of expressivity. These Description Logics (DL) are decidable fragments of first-order logics where explicit knowledge is expressed in axioms and assertions. DL knowledge bases have well-defined model-theoretic semantics that allows to infer new conclusions from existing knowledge.

When ontologies evolve, knowledge is removed or new knowledge is added. By adding new axioms and/or new assertions, contradictions may be introduced

that cause the knowledge base as a whole to be inconsistent. For example, adding disjoint-axioms may make some concepts in the ontology unsatisfiable. When there exists an assertion of some individual to that concept, the knowledge base is inferred to be inconsistent. Yet, for an inconsistent knowledge base any conclusion—even meaningless ones—becomes trivially true. For ontology evolution, it is hence desirable to prevent concepts from being inferred unsatisfiable. There indeed exist more reasons for why a knowledge base can become inconsistent, but *we propose to start off with conflict-free conceptualizations and present a method that never infers any concept to be unsatisfiable.*

Contradictions may be caused by agents that automatically assemble new information from different heterogeneous sources. This is likely to happen when combining different ontologies, modeled by different agents. But even ontologies created by knowledge engineers may contain conflicting pieces of knowledge. As such, *any framework for ontology evolution has to provide a way for resolving conflicts*—usually referred to as *ontology repair*.

Agents querying (or interacting with) an ontology in the Semantic Web assume that both the query and the answer are expressible in OWL2. Furthermore, the answer should have meaningful semantics and should not contain conflicts. We therefore propose to demand, in the context of ontology evolution, any formalism for ontology repair to fulfill the following properties:

1. The formalism for knowledge representation is not changed.
2. No concept is inferred unsatisfiable[4].
3. The procedure shall work automatically.
4. The original information should be kept.
5. As little inferred information as possible shall be lost.

Instead of removing axioms (i.e. explicit knowledge), we propose to invalidate those inferences (i.e. implicit knowledge) that cause concepts to be inferred unsatisfiable. We can show that we lose fewer inferences than in the case of axiom removal. To invalidate only inferences we keep the formalism for knowledge representation. Namely, we consider DL knowledge bases so that the first desired property is satisfied, and apply the following reasoning approach.

Based on Lehmann's Default Logics [2] and Probabilistic Description Logics [3] our approach separates axioms responsible for a conflict into different partitions and collects axioms not involved in a conflict in its separate set. Due to the separate treatment of conflict-causing axioms, inferences for unsatisfiable concepts are invalidated but the formalism for knowledge representation remains unchanged. To limit the number of inferences lost when reasoning the single partitions are considered alongside with the set of non-conflict axioms. We call the pair of the partition and the set of non-conflicting axioms a *Default TBox* which contains all axioms that occurred in the original ontology.

[4] For the current work we concentrate on resolving unsatisfiable concepts. The procedure may, in principle, be extended to resolve unsatisfiable roles and—to a certain extent—assertions.

While this method, unfortunately, cannot prevent that some potentially interesting inferences are lost, it limits the loss to a minimum. It can, furthermore, be shown that fewer inferences are lost than in the case of axiom removal—while still working fully automatically. We indeed accept potential modeling errors in the knowledge base, because there is no evidence that tells us how to solve them otherwise.

This work is structured as follows: We start with an overview over OWL, Description Logics and justifications in Section 2. In the subsequent Section 3 we introduce Default TBoxes and continue in Section 4 with the description of our partition method for invalidating inferences. There we show that the Default TBox resulting from the partition method is consistent and how it can be computed efficiently. We provide experimental results in Section 5 and discuss the findings of our work in Section 6—followed by an overview over related work in the following Section 7. In the final Section 8 we conclude the paper and give an outlook on future work.

2 Preliminaries

Description Logics (DL) are languages for knowledge representation with a well-defined syntax and semantics. A DL allows to specify concepts (also known as classes), instances thereof, also referred to as individuals and binary relations, called roles between individuals. Operators such as negation ($\neg$) or conjunction ($\sqcap$) enable the construction of complex concepts and roles.

A DL allows to define a *terminology of concepts*. Such a terminology, called a *TBox*, is a finite set of axioms describing a hierarchy on a set of concepts. TBox axioms are of the form $Birds \sqsubseteq Animals$ and express, for example, that the concept $Birds$ is a specialization (or subsumed by) of the concept of $Animals$. The semantics of these *subclass-axioms* requires each instance of the concept of $Birds$ also to be an instance of the concept of $Animals$.

Depending on its expressivity, a DL may also be used for defining a role hierarchy, called *RBox*. A finite collection of assertions about individuals is called an *ABox*. A DL *knowledge base* is defined by the triple $K = (T, R, A)$ where T is a TBox, R is an RBox and A is an ABox—each of which is finite and possibly empty.

Interpretations and Satisfiability The formal semantics of a DL is given by an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$. It consists of the *interpretation domain* $\Delta^{\mathcal{I}}$, a non-empty set, and the *interpretation function* $\cdot^{\mathcal{I}}$ which assigns to every atomic concept A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to every atomic role R a set $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

In order to find out whether there exist contradictions in a DL knowledge base $\mathcal{K}$, we have to find out whether there exists an interpretation that satisfies $\mathcal{K}$. An interpretation $\mathcal{I}$ is said to *satisfy* a concept A or a role R, if the result of the interpretation function is not empty. We denote this by $\mathcal{I} \models A$, if $A^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models R$, if $R^{\mathcal{I}} \neq \emptyset$, respectively. Concepts that have an empty interpretation are called *unsatisfiable*, referred to as $\mathcal{T} \models U \sqsubseteq \bot$.

For concept subsumption and role inclusion, satisfiability is defined w.r.t. set inclusion. An interpretation $\mathcal{I}$ is said to satisfy an inclusion axiom $\mathcal{I} \models B \sqsubseteq A$, if $B^{\mathcal{I}} \subseteq A^{\mathcal{I}}$. A TBox $\mathcal{T}$ is said to be satisfiable, if there exists an interpretation $\mathcal{I}$ such that $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$. In this case, we say that $\mathcal{I}$ is a *model* of $\mathcal{T}$. The same holds for RBoxes. In the following, we always assume the RBox to be satisfiable.

2.1 Description Logics Reasoning

A DL knowledge representation system (KRS) defines algorithms to infer implicit knowledge from explicitly stated knowledge. Using these algorithms, a KRS usually supports the following four standard *reasoning* tasks:
Subsumption Test, Satisfiability Test, Consistency Test, and Instance Retrieval.

We say that a TBox $\mathcal{T}$ *entails* an axiom η, or $\mathcal{T} \models \eta$, if *all* models of $\mathcal{T}$ satisfy η. If η is a new axiom, i.e. $\eta \notin \mathcal{T}$, we call η an *inferred axiom*. The *deductive closure* (or extension) of a TBox $\mathcal{T}$ is the set of all non-trivial explicitly stated as well as inferred entailments: $(\mathcal{T})^+ = \{\eta \mid \mathcal{T} \models \eta\}$. Excluding trivial entailments, e.g. $C \sqsubseteq A \sqcap A, C \sqsubseteq A \sqcup B$, we ensure the deductive closure to be finite. We define the deductive closure of an RBox in the same way as for TBoxes.

2.2 Description Logics and OWL

Knowledge in the Semantic Web is represented by ontologies which are expressed in the *Web Ontology Language* (OWL). The current standard OWL2 [1] was designed in a way such that it has the DL $\mathcal{SROIQ}$ as a rough syntactic variant. $\mathcal{SROIQ}$ is a very expressive language, and its reasoning procedures are of high computational complexity. Yet, this expressiveness is not needed in every case. As a consequence, so-called profiles were defined for OWL2, each of which correspond to a unique DL. Hence, every OWL2 profile has a well-defined semantics and a decidable reasoning procedure. The profiles were defined in a way such that the knowledge representation is unchanged but language constructors are restricted in a way such that the corresponding DL is less expressive.

Thanks to the general nature of the method proposed in this paper we may consider any DL for which there exists a decidable reasoning procedure. In other words: we are not dependent on the expressiveness of the OWL2 profile used.

2.3 Justifications

To *invalidate* an unwanted inference $\mathcal{T} \models \eta$, i.e. removing η from $(\mathcal{T})^+$, we must first find out, which axioms are responsible for this very inference. We are hence looking for the minimal sets of axioms $J \subseteq \mathcal{T}$ for which $J \models \eta$ holds [4]. Minimality guarantees that the entailment is invalidated when at least one of the axioms from the corresponding minimal set is not considered alongside with the others in the $\mathcal{T}$ when drawing inferences. Justifications are minimal sets of axioms from a TBox such that an inference still holds:

Definition 1. *Let $\mathcal{T}$ be a TBox. A justification for an entailment $\mathcal{T} \models \eta$ is a set of axioms $J_{\mathcal{T},\eta} \subseteq \mathcal{T}$ such that $J_{\mathcal{T},\eta} \models \eta$ and $J' \not\models \eta$ for every $J' \subset J_{\mathcal{T},\eta}$.*

In order to be able to distinguish individual justifications, we number them by $k = 0, \ldots, K$ where K is the number of justifications. Consequently, refer to the k-th justification by $J^k_{\mathcal{T},\eta}$.

Justifications for unsatisfiable concepts, called *unsat justifications*, are of the form $J_{\mathcal{T},U\sqsubseteq\bot}$ where U is a (possibly complex) unsatisfiable concept. Unsat justifications may depend completely on the satisfiability of other concepts, i.e. they are supersets of other justifications. If an unsat justification does not depend on another one, then we call it a *root unsat justification*.

Definition 2. *Let $\mathcal{T}$ be a TBox. A justification for $J_{\mathcal{T},U\sqsubseteq\bot}$ is called a* derived unsat justification*, if there exists some concept U' for which $J_{\mathcal{T},U\sqsubseteq\bot} \supset J_{\mathcal{T},U'\sqsubseteq\bot}$. Otherwise $J_{\mathcal{T},U\sqsubseteq\bot}$ is called a* root unsat justification*.*

By invalidating some unsat justification $J_{\mathcal{T},U\sqsubseteq\bot}$, i.e. by performing some operation such that $J_{\mathcal{T},U\sqsubseteq\bot} \not\models U \sqsubseteq \bot$, all dependent unsat justifications are invalidated as well. Hence, invalidating all root unsat justifications for unsatisfiable concepts will make every so far unsatisfiable concept satisfiable again. Referring to the properties we proposed in Section 1, a framework for conflict-free ontology evolution has to provide a method for automatically invalidating all root unsat justifications for unsatisfiable concepts.

Example 1. Given the simple TBox $\mathcal{T} = \{B \sqsubseteq A, C \sqsubseteq B, C \sqsubseteq \neg A, D \sqsubseteq C\}$ it is possible to infer the concept C to be unsatisfiable. The only justification for $\mathcal{T} \models U \sqsubseteq \bot$ is $J_{\mathcal{T},C\sqsubseteq\bot} = \{B \sqsubseteq A, C \sqsubseteq B, C \sqsubseteq \neg A\}$. The concept D is also inferred to be unsatisfiable, but with the justification $J_{\mathcal{T},D\sqsubseteq\bot} = J_{\mathcal{T},C\sqsubseteq\bot} \cup \{D \sqsubseteq C\}$. As such, $J_{\mathcal{T},C\sqsubseteq\bot}$ is a root unsat justification whereas $J_{\mathcal{T},D\sqsubseteq\bot}$ is derived.

3 Invalidating Inferences

In this Section, we define a general operator for invalidating inferences. After introducing the formal basics that underly our method for solving conflicts we show how it can be applied to the general operator.

3.1 A General Operator for Invalidating Inferences

Let $\mathcal{T}$ be a TBox and $\eta \in (\mathcal{T})^+$. Applying some operator $\Delta_{(\cdot)}$ to $\mathcal{T}$, we say the inference $\mathcal{T} \models \eta$ is invalidated, if η is not contained in the the deductive closure of the result of $\Delta_{\mathcal{T}}$, i.e. $(\eta \notin (\Delta_{\mathcal{T}})^+)$. As stated in Section 1, we require the knowledge representation to be unchanged, resulting the deductive closure $(\Delta_{\mathcal{T}})^+$ to be a set of DL axioms as well.

There exist various ways of defining the $\Delta_{(\cdot)}$ operator, like para-consistent logics [5], and removing axioms from $\mathcal{T}$ that justify $\mathcal{T} \models \eta$. The latter is the currently most common way for solving conflicts and usually referred to as ontology

repair. Instead of actually removing an explicitly stated axiom, we propose to split up the TBox such that groups of axioms causing unsatisfiability are not used at the same time when drawing inferences. That way it is still possible to reason, even if the KB is unsatisfiable. We keep the formalism for knowledge representation but have to slightly change the way of inferencing. In order to do this the notion of *Default Knowledge Bases* is introduced in the following.

3.2 Default Knowledge Bases

Inference services like Lehmann's Lexicographical Entailment for Default Logics [2] provide exactly the desired properties from Section 1. To achieve these we split the set of axioms into a TBox $\mathcal{T}_\Delta$ and partitions such that for each partition all concepts of all axioms in the partition together with the axioms in $\mathcal{T}_\Delta$ are satisfiable. We achieve a family of TBoxes $(\mathcal{T}_\Delta \cup \mathcal{U}_n)_{n=0}^N$ which we call a *Default TBox*. Inferences are invalidated by treating the partitions of axioms separately.

Definition 3. *Let $\mathcal{T}_\Delta$ and $\mathcal{U}_0, \ldots, \mathcal{U}_N$ be mutually disjoint TBoxes. A Default TBox $\mathcal{DT}$ is the family of TBoxes $\mathcal{DT} = (\mathcal{T}_\Delta \cup \mathcal{U}_0, \ldots, \mathcal{T}_\Delta \cup \mathcal{U}_N)$.*

A Default TBox implies a partitioning on a TBox: $\mathcal{T} = \mathcal{T}_\Delta \oplus \mathcal{U}_0 \oplus \ldots \oplus \mathcal{U}_N$. According to [2] and [3], a single partition $\mathcal{U}_n$ is defined as the set of all axioms $B \sqsubseteq A$ in $\mathcal{T} \setminus (\mathcal{T}_\Delta \bigcup_{m=0}^{n-1} \mathcal{U}_m)$ for which $\mathcal{T}_\Delta \bigcup_{m=0}^{n-1} \mathcal{U}_m \not\models A \sqcap B$. As a result, $\mathcal{T}_\Delta \cup \mathcal{U}_n$ does not infer any concept to be unsatisfiable. This enables us to define a Δ-operator such that the partitioning of a TBox is transformed into a Default TBox:

Definition 4. *Let $\mathcal{T}$ be a TBox and $\mathcal{U}_0, \ldots, \mathcal{U}_N$ be a family of disjoint subsets of $\mathcal{T}$. Let further be $\mathcal{T}_n = \mathcal{T}_\Delta \cup \mathcal{U}_n$, and $\mathcal{T}_\Delta = \mathcal{T} \setminus \bigcup_{n=0}^N \mathcal{U}_n$. The Default Δ-operator transforms $\mathcal{T}$ and $\mathcal{U}_0, \ldots, \mathcal{U}_N$ to a Default TBox:*

$$\Delta_\mathcal{T}(\mathcal{U}_0, \ldots, \mathcal{U}_N) = \mathcal{DT} \text{ with } \mathcal{DT} = (\mathcal{T}_n)_{n=1}^N$$

3.3 Inferences for a Default TBox

A Default TBox is a wrapper for a set of classical TBoxes. Hence, inference services like checking axiom satisfiability and consistency can be applied by means of the inference service defined for the original TBox. Consequently, we introduce the following definitions:

A Default TBox $\mathcal{DT}$ is said to entail an axiom, $\mathcal{DT} \models \eta$, if one of its TBoxes entails η. As a result, the deductive closure of a Default TBox is defined by $(\mathcal{DT})^+ = \bigcup_{n=0}^N (\mathcal{T}_n)^+$. We say that $\mathcal{DT}$ is satisfiable if all of its TBoxes are satisfiable.

Please note that this inference service for a Default TBox is not the same inference service as defined by Lehmann's Lexicographical Entailment: We borrowed the method of how to separate axioms from each other into partitions but we compute regular models instead of lex-minimal models. The actual partitioning process is described in Section 4.

3.4 Inferences Invalidated

As we consider axioms separately during reasoning, we will lose inferences. In the sequel, we show which inferences are actually lost by separating axioms into different partitions.

Compared to the original TBox, in the Default TBox all inferences whose justifications share axioms that are contained in different partitions are invalidated:

$$(\mathcal{T})^+ \setminus (\mathcal{DT})^+ = \{\ \eta \quad | \ J_{\mathcal{T},\eta} \cap \mathcal{U}_n \neq \emptyset \quad \wedge \quad J_{\mathcal{T},\eta} \cap \mathcal{U}_m \neq \emptyset \quad \wedge \quad n \neq m\}$$

In Example 1, a valid Default TBox was $\mathcal{T}_\Delta = \{D \sqsubseteq C\}$ and $\mathcal{U}_0 = \{B \sqsubseteq A\}$ and $\mathcal{U}_1 = \{C \sqsubseteq B, C \sqsubseteq \neg A\}$. While the bad inferences $C \sqsubseteq \bot$ and $D \sqsubseteq \bot$ are invalidated, also $C \sqsubseteq A$ and $D \sqsubseteq A$ are not valid anymore. On the other hand, for the Default TBox $\mathcal{T}_\Delta = \{C \sqsubseteq B, D \sqsubseteq C\}$ and $\mathcal{U}_0 = \{B \sqsubseteq A\}$ and $\mathcal{U}_1 = \{C \sqsubseteq \neg A\}$ these inferences are preserved.

If we applied classical ontology repair, i.e. axiom removal, not only we delete the removed axioms from the deductive closure of the TBox, but even inferences whose justifications contain just one of the removed axiom are invalidated—in contrast to our method where we only invalidate justifications containing pairs of axioms that are in different partitions. We hence claim that using our method for defining the Δ-operator, it can be expected that fewer inferences are invalidated.

4 Partitions from Minimal Unsatisfiability Splitting

To resolve unsatisfiable concepts, we split up the sets of trouble-causing axioms, i.e. the root justifications for unsatisfiable concepts, into different partitions. This is achieved by applying the adapted version of the partition method from Lehmann's Lexicographical Entailment [2] as defined for Probabilistic Description Logics [3] that we introduced in [6].

In [6] we showed that under certain conditions we can compute a valid partition consisting of all the axioms of the root unsat justifications without additional satisfiability checks. This methods indeed defines a Δ-operator for invalidating all troublesome inferences $\mathcal{T} \models U \sqsubseteq \bot$. We here show that we can improve this Δ-operator by putting only exactly two axioms of each root unsat justification into the partitions whereas the remaining axioms can be put in the TBox. Fewer axioms in the partitions, in turn, means potentially loosing fewer inferences.

We will not change the working principle of the procedure introduced in [6]. As a result, the improved procedure will also be a valid Δ-operator. In the following we show how the splitting can be used for computing the partitions. We show by induction which axioms of the root justifications we have to chose for the partitions $\mathcal{U}_n$of the Default TBox and which we may leave for $\mathcal{T}_\Delta$.

4.1 Unsatisfiability Splitting

We split-up any unsat justification $J^k_{\mathcal{T},U\sqsubseteq\bot}$ into the *splitting sets* [6]:

$$\Gamma^k_{U\sqsubseteq\bot} = \{U \sqsubseteq A \in J^k_{\mathcal{T},U\sqsubseteq\bot}\} \text{ and } \Theta^k_U = J^k_{\mathcal{T},U\sqsubseteq\bot} \setminus \Gamma^k_U$$

We refer to axioms from the first as γ-axioms whereas we call axiom from the latter θ-axioms.

Example 2. For the TBox $\mathcal{T} = \left\{ \begin{array}{lllllllll} (1) & B & \sqsubseteq & \neg A & (2) & \exists R.\top & \sqsubseteq & A & (3)\ C \sqsubseteq B \\ (4) & D & \equiv & B \sqcap \forall R.\{p\} & (5) & E & \sqsubseteq & \neg D & (6)\ F \sqsubseteq D \\ (7) & H \sqcap J & \sqsubseteq & \neg B \sqcap F & (8) & G & \sqsubseteq & H \sqcap J & \end{array} \right\}$

all justifications for all unsatisfiable concepts can be split up into Θ and Γ as follows:

$$\begin{array}{llll} & \overbrace{\quad\Theta\quad} & \overbrace{\Gamma} & \\ J^1_{\mathcal{T},D\sqsubseteq\bot} = & \{(1),(2)\} & \cup\ \{(4)\} & \\ J^3_{\mathcal{T},F\sqsubseteq\bot} = & \{(1),(2),(4)\} & \cup\ \{(6)\} & \\ J^5_{\mathcal{T},G\sqsubseteq\bot} = & \{(4),(6),(7)\} & \cup\ \{(8)\} & \end{array} \qquad \begin{array}{llll} & \overbrace{\quad\Theta\quad} & \overbrace{\Gamma} & \\ J^2_{\mathcal{T},H\sqcap J\sqsubseteq\bot} = & \{(4),(6)\} & \cup\ \{(7)\} & \\ J^4_{\mathcal{T},H\sqcap J\sqsubseteq\bot} = & \{(1),(2),(4),(6)\} & \cup\ \{(7)\} & \end{array}$$

The separation of a justification into Θ and Γ provides exactly the separation we seek for computing the partitions: for each root unsat justification exactly one θ-axiom is placed into partition $\mathcal{U}_n$ and exactly one γ-axiom is placed into partition $\mathcal{U}_{n+1}$. The remaining axioms are finally added to $\mathcal{T}_\Delta$.

4.2 Improved Partitioning Algorithm

Algorithm 1 describes the improved procedure for finding partitions from the splitting. In the sequel we will illustrate its working principles by Example 2.

Initializing $\mathcal{T}_\Delta$ For all axioms $B \sqsubseteq A$ that do not occur in any root justification, the requirement $A \sqcap B$ to be satisfiable is trivially fulfilled. Hence, we initialize $\mathcal{T}_\Delta$ with $\mathcal{T}_\Delta = \mathcal{T} \setminus \bigcup_k^K J^k_{\mathcal{T},U\sqsubseteq\bot}$. In Example 2, the only axioms that are not part of any root justification are (3), (5), and (8). We hence start with $\mathcal{T}_\Delta = \{(3),(5),(8)\}$.

Computing a Partition For computing a partition, we consider all those root justifications for which neither of their Θ-axioms occurs in a Γ-set of *any* other unsat justification (Line 15).

In Example 2, axioms (4), (6) and (7) are in some Γ-set and are therefore not candidates for the first partition $\mathcal{U}_0$. We may only choose one axiom from $\{(1),(2)\} \subset \Theta^1_{D\sqsubseteq\bot}$. Assume we choose axiom (2) for $\mathcal{U}_0$ and add axiom (1) to $\mathcal{T}_\Delta$.

We resolved $\Theta^1_{D\sqsubseteq\bot}$, and remove it from the splitting sets (Line 18). We take exactly one axiom from its Γ-set and add it to partition $\mathcal{U}_1$ (Line 6). In this case, $\Gamma^1_{D\sqsubseteq\bot} = \{(4)\}$, so the only choice is axiom (4). We resolved the root unsat

Algorithm 1 The partitioning algorithm.

Input:

1. TBox $\mathcal{T}$,
2. Split-up unsat justifications $\mathcal{J} = \{J^k_{\mathcal{T},U\sqsubseteq\bot}\}$ for $k = 0, \ldots, K$

Output:

Default TBox $\mathcal{DT} = (\mathcal{T}_\Delta \cup \mathcal{U}_0, \ldots, \mathcal{T}_\Delta \cup \mathcal{U}_N)$

1: // Initialization
2: $\mathcal{T}_\Delta \leftarrow \mathcal{T} \setminus \bigcup_{k=0}^{K} J^k_{\mathcal{T},U\sqsubseteq\bot}$, $\quad \Theta \leftarrow \{\Theta^k_{U\sqsubseteq\bot}\}_{k=1}^{K}$, $\quad \Gamma \leftarrow \{\Gamma^k_{U\sqsubseteq\bot}\}_{k=1}^{K}$, $\quad n \leftarrow 0$
3: **while** $\Theta, \Gamma \neq \emptyset$ **do**
4: $\quad \mathcal{U}_n \leftarrow \emptyset$
5: $\quad$ // Solve conflicts for all Γ-sets whose Θ-set was removed
6: $\quad$ **for all** $\Gamma^k_{U\sqsubseteq\bot}$ with $\Theta^k_{U\sqsubseteq\bot} \notin \Theta$ **do**
7: $\quad\quad \mathcal{U}_n \leftarrow \mathcal{U}_n \cup \{\gamma\}$ for exactly one $\gamma \in \Gamma^k_{U\sqsubseteq\bot}$
8: $\quad\quad \mathcal{T}_\Delta \leftarrow \mathcal{T}_\Delta \cup (\Gamma^k_{U\sqsubseteq\bot} \setminus \mathcal{U}_n)$
9: $\quad\quad$ // Delete all resolved Γ-sets.
10: $\quad\quad \Gamma \leftarrow \Gamma \setminus \Gamma^k_{U\sqsubseteq\bot} \cup \{\Gamma^{k'}_{U'\sqsubseteq\bot} | J^{k'}_{\mathcal{T},U'\sqsubseteq\bot} \supset J^k_{\mathcal{T},U\sqsubseteq\bot}\}$
11: $\quad\quad$ // Delete all resolved Θ-sets.
12: $\quad\quad \Theta \leftarrow \Theta \setminus \{\Theta^{k'}_{U'\sqsubseteq\bot} | J^{k'}_{\mathcal{T},U'\sqsubseteq\bot} \supset J^k_{\mathcal{T},U\sqsubseteq\bot}\}$
13: $\quad$ **end for**
14: $\quad$ // Resolve Θ-sets having axioms not contained in any Γ-set
15: $\quad$ **for all** $\Theta^k_{U\sqsubseteq\bot}$ with $\Theta^k_{U\sqsubseteq\bot} \setminus \bigcup_{k'=0}^{K} \Gamma^{k'}_{U'\sqsubseteq\bot} \neq \emptyset$ **do**
16: $\quad\quad \mathcal{U}_n \leftarrow \mathcal{U}_n \cup \{\theta\}$ for exactly one $\theta \in \Theta^k_{U\sqsubseteq\bot} \setminus \bigcup_{k'=0}^{K} \Gamma^{k'}_{U'\sqsubseteq\bot}$
17: $\quad\quad \mathcal{T}_\Delta \leftarrow \mathcal{T}_\Delta \cup \left(\Theta^k_{U\sqsubseteq\bot} \setminus (\mathcal{U}_n \cup \bigcup_{k'=0}^{K} \Gamma^{k'}_{U'\sqsubseteq\bot})\right)$
18: $\quad\quad \Theta \leftarrow \Theta \setminus \Theta^k_{U\sqsubseteq\bot}$ $\quad$ // remove solved Θ-sets.
19: $\quad$ **end for**
20: $\quad n \leftarrow n + 1$
21: **end while**

justification $J^1_{\mathcal{T},D\sqsubseteq\bot}$, as well as the dependent unsat justifications $J^3_{\mathcal{T},F\sqsubseteq\bot}$ and $J^4_{\mathcal{T},H\sqcap J\sqsubseteq\bot}$, all of whose splitting sets we remove (Lines 10, 12).

Axiom (4) was also part of $\Theta^2_{H\sqcap J\sqsubseteq\bot}$. Hence we resolved $\Theta^2_{H\sqcap J\sqsubseteq\bot}$ which means that we may add its remaining axioms, i.e. axiom (6) to $\mathcal{T}_\Delta$. Finally, we remove this splitting set (Line 18).

We again resolve its corresponding Γ-set which contains only axiom (7). Since we now resolved all root justifications, also the remaining depending justification $J^5_{\mathcal{T},G\sqsubseteq\bot}$ is resolved. Hence, there are no more Θ-sets to process, and the procedure terminates.

4.3 Consistency of the Default TBox

When the procedure terminates, all dependent unsat justifications have been resolved, i.e. the Default TBox $\mathcal{DT} = \Delta_\mathcal{T}(\mathcal{U}_0, \ldots, \mathcal{U}_N)$ does not infer any of its concepts to be unsatisfiable. Since the partitions $\mathcal{U}_n$ were constructed according to [3], for any choice of the θ-axioms, $\mathcal{DT}$ is consistent. Note that the actual

number of partitions and their axioms depend on the choice of θ-axioms, but are unique for each choice. This also means that each choice may invalidate different sets of inferred entailments and, as a result, the impact of the Default Δ-operator depends on the actual choice made.

In Example 2, the Default TBox is determined by $\mathcal{T}_\Delta = \{(1), (3), (5), (6), (8)\}$ with the partitions $\mathcal{U}_0 = \{(2)\}$, $\mathcal{U}_1 = \{(4)\}$, and $\mathcal{U}_2 = \{(7)\}$.

4.4 Complexity

The complexity of the whole procedure is dominated by finding all unsat justifications which is NEXPTIME-complete in the size of the TBox. Finding partitions does not require any reasoning but only set operations which is worst-case quadratic in the number of axioms in the unsat justifications. Hence, our approach has the same worst-case time complexity as ontology repair, which also relies upon finding all unsat justifications.

5 Experimental Results

To evaluate the presented approach, we performed experiments on three ontologies that are known to contain unsatisfiable concepts. We compared the deductive closure of the original TBox with (1) all possible repair solutions, (2) the approach where all axioms of all root unsat justifications are put into partitions and (3) the presented approach. Since the number of possible repair solutions grows exponentially with the number of axioms in the root unsat justifications we restricted the experiments to rather small ontologies.

5.1 Experiment Layout

If an ontology contains an unsatisfiable concept, depending on the reasoner, the concept hierarchy for unsatisfiable concepts might be undefined. Hence, we compute an approximation of the deductive closure of the original TBox (containing unsatisfiable concepts) as the union of the deductive closures of all TBoxes $\mathcal{T}_r$ that result from all possible repair solutions. For comparison, we took the repair solution where the minimal number of inferences is lost.

We computed the deductive closure for each of the possible Default TBoxes that can be obtained using the presented approach. For comparison with deductive closure of the original TBox and the best repair solution, we took that Default TBox solution $\mathcal{DT}_{sgl}$ where the minimal number of inferences is lost.

Furthermore, we computed the deductive closure for the Default TBox $\mathcal{DT}_{all}$ where we put all axioms from all root unsat justifications into partitions. Since its solution is unique, we can use it directly for comparison with the deductive closure of the original TBox and the best repair solution.

For computing justifications we used the black-box approach described in [7]. All experiments were performed using the Pellet OWL2 reasoner [8], version 2.2.0, and the Manchester OWL API [9], version 3.0.

	$\lvert(\mathcal{T})^+\rvert$	$\lvert(\mathcal{T}_r)^+\rvert$	$\lvert(\mathcal{DT})^+\rvert$		$\lvert(\mathcal{T}_r)^+ \setminus (\mathcal{DT})^+\rvert$		$\lvert(\mathcal{DT})^+ \setminus (\mathcal{T}_r)^+\rvert$	
			all	*sgl*	*all*	*sgl*	*all*	*sgl*
Koala	68	68	68	86	1	0	1	18
Chemical	293	261	233	293	61	0	33	33
Pizza	1151	1151	1150	1152	1	0	2	1

Table 1. Comparison of the deductive closure of the original TBox ($\mathcal{T}$) with the best solution of all possible repair solutions ($\mathcal{T}_r$), the TBox of the approach of putting all axioms of all root unsat justifications into partitions ($\mathcal{DT}_{sgl}$) and the best solution of all possible Default TBoxes of the approach presented ($\mathcal{DT}_{all}$).

5.2 Results

As can be seen from Table 1, the deductive closure of the best solution of our approach $\mathcal{DT}_{sgl}$ (column 4) is, for all ontologies we tested, larger than the deductive closure of the best solution for axiom removal $\mathcal{T}_r$ (column 2). This, in turn, means that the number of entailments invalidated is always lower comparing the best solution of our approach with the best solution of removing axioms. Furthermore, we outperform axiom removal in the sense that the best solution of axiom removal does not preserve any inference different from our approach (column 6).

However, this is not true for the original version of our approach. If we put all axioms of all root unsat justifications into the partitions of the Default TBox $\mathcal{DT}_{all}$ (column 3), we invalidate more inferences than than both the improved version and even the best solution of axiom removal do. However, the actual inferences invalidated differ for the original approach and the best removal solution (column 5). For the Koala and the Pizza ontology the difference is caused by the fact that, w.r.t. the original TBox $\mathcal{T}$, axioms were actually removed from $\mathcal{T}_r$ but not from $\mathcal{DT}_{all}$. In contrast to that, the set of inferences invalidated differs in more than the removed axioms for the more complex Chemical ontology.

6 Discussion

The experimental results support our claim that using our approach we can find a solution which invalidates fewer inferences than using classical axiom removal for ontology repair. We, furthermore, showed that the presented more fine-grained approach indeed improves its original version.

Neither axiom removal nor our approach make a deterministic choice w.r.t. any performance measure that takes into account the number of inferences a choice invalidates. On the one hand, computing all possible solutions is not feasible for large ontologies, on the other hand, the computation of a single valid solution is cheap. Hence, optimization strategies like a stochastic search seem promising for finding optimal solutions.

Our approach is restricted to solving conflicts for unsatisfiable concepts. In an OWL2 knowledge base, however, roles may become unsatisfiable and inconsistencies (i.e. assertions that cause the ABox not to be a model of the TBox)

may occur. Since OWL2, the treatment of roles has become similar to the treatment of concepts. Hence, the approach of splitting sets of axioms that support a role conflict in the TBox may also be applied to unsatisfiable roles. To address inconsistencies we may use abduction to add a new concept for assertions causing an inconsistency. Alternatively, we could treat instances like concepts and apply our approach directly.

We only tested our approach on a few rather small knowledge bases. The complexity of the whole procedure we presented is dominated by the computation of all root justifications for all unsatisfiable concepts. However, when we evolve a knowledge base, incremental reasoning [10] may significantly reduce the computation time. If there was a way how to maintain the root unsat justifications for a TBox, our procedure, and ontology-repair in general, computing all root unsat justifications becomes feasible even for large Semantic Web knowledge bases.

7 Related Work

In recent years, much progress has been made in the task to explain why a conclusion can be drawn from a DL knowledge base by solely using axioms from the knowledge base itself. Schlobach and Cornet [4] came up with minimal unsatisfiable preserving sub-TBoxes (MUPS) which can explain the reason for unsatisfiability of concepts. Kalyanpur et al. [11] introduced justification as a form of minimal explanation for any arbitrary entailment. It could be shown that computing all justifications for an entailment is feasible in the tableaux calculus [11]. Recent approaches try to compute fine-grained [12] or laconic justifications [13] to consider only the conflict causing sub-parts of an axiom.

In the area of conflict-free ontology evolution, the main focus usually lies on resolving inconsistencies and hence changes mainly occur on instance level or rather restricted TBoxes [14]. Repair can also be done using higher-order logics like in the Ontology Repair System [15]. This, however, makes changes to the knowledge representation and cannot be applied to OWL ontologies in a straightforward way.

Alternatives to do reasoning with incoherent DL knowledge bases are, for example, para-consistent logics [5]. However, these change the notion of inference and hence their semantics much more than default logic does.

Default Logics were first introduced by Reiter [16]. Because of undecidability issues Lehmann provided a simpler perspective on Default Logics [2] introducing the concept of partitions. Lukasiewicz extended this approach by a separate TBox that contains all the axioms which still model crisp and not default knowledge [3]. He also enriched defaults by belief intervals resulting in a probabilistic variant of the DL $\mathcal{SHOIN}(\mathbf{D})$ referred to as $P-\mathcal{SHOIN}(\mathbf{D})$ or Probabilistic Description Logics. While in this paper we make use of the partition approach enriched by a TBox, we do not consider the possibility of assigning the axioms belief intervals—although this should, in principle, be possible.

There have been made propositions of how to incorporate default knowledge in OWL-DL knowledge bases in [17] [18], and [19]. While the first two deal with applications of Reiter's interpretation of defaults, to our knowledge, P-$\mathcal{SHOIN}(\mathbf{D})$ [3] is currently the only formalism providing default reasoning services w.r.t. Lehmann's lexicographical entailment for OWL DL knowledge bases for which an implementation is available [20].

8 Conclusion and Outlook

In this paper, we presented an approach for automatic conflict-free ontology evolution. We defined a general framework and proposed a variant of Lehmann's Default Logics and Probabilistic Description Logics that relies upon the idea of considering conflict-causing axioms separately when drawing inferences. In contrast to classical ontology repair, we proposed to keep the explicitly stated axioms but remove only implicit inferences for solving conflicts. This separation can be achieved by applying a splitting scheme to the root justifications for conflicts, which are, in our case, unsatisfiable concepts. The partition and the remaining axioms form a Default TBox, which slightly changes the inferencing procedure but leaves the knowledge representation untouched. The deductive closure of the automatically constructed Default TBox was shown to be free of conflicts. Experimental results confirmed that applying our method fewer inferences were lost than in the case of classical automatic ontology repair.

However, we restricted our experiments to rather small knowledge bases compared to what is currently considered as a large knowledge base. The main reason for this limitation is the computational complexity of the generation of justifications for all unsatisfiable concepts. If it were possible to efficiently maintain justifications for evolving knowledge bases, the approach could also be applied to large scale knowledge bases.

The choice of the actual partition for the Default TBox is non-deterministic. For large scale knowledge bases, computing solutions is likely to become unfeasible. Hence, optimization strategies, like a stochastic search process, may be applied to look for an optimal solution. Such optimization strategies may also take into account more sophisticated performance measures like the impact of invalidated inferences instead of just counting their total number.

Since current approaches to the computation of fine-grained justifications increasingly aim to only address sub-parts of the axioms, future work will investigate how parts of axioms may be used for a Default TBox. Last, we restricted conflicts to unsatisfiable concepts. Since the presented approach is rather generic, it should, in general, be applicable to unsatisfiable roles and inconsistencies as well.

As sketched above, all these limitations do not invalidate the general approach but are the subject of future work. Indeed we believe that ontology repair using the presented approach has the potential to enable automatic conflict-free ontology evolution in practice—a central need for long living ontologies and, hence, for the Semantic Web.

References

1. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S.: OWL 2 Web Ontology Language Primer. W3C Recommendation, World Wide Web Consortium (October 2009)
2. Lehmann, D.: Another perspective on default reasoning. Ann. Math. Artif. Intell **15** (1995) 61–82
3. Lukasiewicz, T.: Expressive probabilistic description logics. Art. Intell. **172**(6-7) (April 2008) 852–883
4. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: IJCAI. (2003) 355–362
5. Ma, Y., Lin, Z., Lin, Z.: Inferring with inconsistent owl dl ontology: A multi-valued logic approach. In: EDBT Workshops. Volume 4254 of LNCS. (March 2006) 535–553
6. Scharrenbach, T., Grütter, R., Waldvogel, B., Bernstein, A.: Structure Preserving TBox Repair using Defaults. In: Proc. of the 23rd International Workshop on Description Logics (DL 2010). CEUR-ws (2010)
7. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in owl ontologies. Journal of Web Semantics **3**(4) (December 2005) 268–293
8. Sirin, E., Parsia, B., Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Web Semantics: Science, Services and Agents on the World Wide Web **5**(2) (June 2007) 51–53
9. Horridge, M., Bechhofer, S.: The owl api: A java api for working with owl 2 ontologies. In: OWLED. Volume 529. (2008)
10. Parsia, B., Halaschek-wiener, C., Sirin, E.: Towards incremental reasoning through updates in owl dl. In: WWW '05: Proc. of the 15th international conference on World Wide Web. (2006)
11. Kalyanpur, A.: Debugging and Repair of OWL Ontologies. PhD thesis, University of Maryland, Department of Computer Science (2006)
12. Lam, J.S.C., Sleeman, D., Pan, J.Z., Vasconcelos, W.: A fine-grained approach to resolving unsatisfiable ontologies. (2008) 62–95
13. Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in OWL. In: The Semantic Web - ISWC 2008. Volume 5318 of LNCS. (October 2008) 323–338
14. Haase, P., Völker, J.: Ontology learning and reasoning - dealing with uncertainty and inconsistency. In: Uncertainty Reasoning for the Semantic Web 1: ISWC International Workshop, URSW 2005-2007, Revised Selected and Invited Papers: Pt. 1. Volume 5327 of LNCS. (January 2009) 45–55
15. Bundy, A.: Where's my stuff? an ontology repair plan. In: Workshop on DISPROVING - Non-Theorems, Non-Validity, Non-Provability. Volume 4. (July 2007) 2–11
16. Reiter, R.: A logic for default reasoning. Art. Intell. **13**(1-2) (1980) 81–132
17. Baader, F., Hollunder, B.: Embedding defaults into terminological knowledge representation formalisms. In: Journal of Automated Reasoning. (1995) 306–317
18. Dao-Tran, M., Eiter, T., Krennwallner, T.: Realizing default logic over description logic knowledge bases. In: ECSQARU 2009. 602–613
19. Navarro, J.L., Sanchez, J.M., Zurita, J.M.: Default reasoning in web ontology language. In: Proc. IADIS Multi Conf. on Computer Science and Information Systems 2007. (July 2007) 35–42
20. Klinov, P.: Pronto: A non-monotonic probabilistic description logic reasoner. In: ESWC. Volume 5021 of Lecture Notes in Computer Science. (2008) 822–826