Semantically Guided Evolution of $\mathcal{SHI}$ ABoxes

Ulrich Furbach and Claudia Schon

University of Koblenz-Landau, Germany, email:{uli,schon}@uni-koblenz.de

Abstract This paper presents a method for the evolution of $\mathcal{SHI}$ ABoxes which is based on a compilation technique of the knowledge base. For this the ABox is regarded as an interpretation of the TBox which is close to a model. It is shown, that the ABox can be used for a semantically guided transformation resulting in an equisatisfiable knowledge base. We use the result of this transformation to efficiently delete assertions from the ABox. Furthermore, insertion of assertions as well as repair of inconsistent ABoxes is addressed. For the computation of the necessary actions for deletion, insertion and repair, the E-KRHyper theorem prover is used.

1 Introduction

Description Logic knowledge bases consist of two parts: the TBox and the ABox. The TBox contains the terminological knowledge and describes the world using so called concepts and roles. The ABox contains knowledge about individuals, stating to which concepts they belong to and via which roles they are connected. There is a considerable amount of work introducing update algorithms and mechanisms for Description Logic knowledge bases, which is of great interest to the Semantic Web community (see [11,16] for details). It is an indisputable fact, that in practice, knowledge bases are subject to frequent changes ([10]) and that even the construction of a knowledge base can be seen as an iterative process. On the other hand this abets inconsistencies in knowledge bases. Therefore the removal of inconsistencies from knowledge bases is of great interest as well ([13]). In this paper we are interested in an evolution of the knowledge base on the instance level. For this, we consider the TBox to be fixed and consistent. We address three different operations on the instance level of the knowledge base: *deletion*, *insertion* and *repair*. Instance-level deletion means the deletion of an instance assertion from the deductive closure or the knowledge base by removing as few assertions as possible. Instance-level insertion means adding an instance assertion to the knowledge base. In both cases it is important that the resulting knowledge base is consistent. For the task of ABox repair we are given an inconsistent knowledge base with consistent terminological part. The aim is to remove assertions from the ABox such that the resulting ABox together with the TBox is consistent. In all three tasks the changes performed should be minimal. This corresponds to the goal of maintaining as much from the original ABox as possible. This view of minimal change corresponds to a *formula based* approach as opposed to a *model based* approach as investigated in [16]. In the model based

approach the set of models of the knowledge base resulting form a change operation should be as close as possible to the set of models of the original knowledge base.

In [15], [7] and [5] instance level deletion, insertion and repair are addressed for DL-Lite knowledge bases. In [14] inconsistent DL-Lite ABoxes are considered. [14] establishes inconsistency-tolerant semantics in order to be able to use those inconsistent ABoxes for query answering. [19] studies the complexity of reasoning under inconsistent-tolerant semantics. Algorithms for the calculation of minimal repair of DL-Lite ABoxes suggested in [14] test the satisfiability of every single ABox assertion and every pair of ABox assertions w.r.t. the TBox. Since for DL-Lite the satisfiability test is tractable, this approach is reasonable. However the ExpTime completeness of consistency testing of $\mathcal{SHI}$ ABoxes forbids such an approach. Further the algorithms suggested in [14] cannot be used for $\mathcal{SHI}$ ABoxes, because these algorithms exploit the following nice property of DL-Lite: as shown in [5], in DL-Lite the unsatisfiability of an ABox w.r.t. a TBox is either caused by a single assertion or a pair of assertions. However in $\mathcal{SHI}$ an arbitrary number of assertions can cause unsatisfiability w.r.t. a TBox.

Our approach is motivated by the observation that a consistent ABox can be seen as a (partial) model of the TBox, which can be used to guide the reasoning process, as proposed in [6]. In [3] this approach was used for model-based diagnosis, where an initial interpretation, which is very close to a model, was used to compute the deviations of a minimal model to this interpretation. In [1] the same approach was applied to view deletion in databases. In our case it is reasonable to assume, that the ABox is very close to a model of the TBox. We use this assumption to semantically guide the construction of instance-based deletion, insertion and repair of ABoxes. As in [3], we gradually revise the assumption of the given ABox being a model for the TBox. This leads to a natural construction of *minimal* instance deletions/insertions and repairs of ABoxes.

The advantage of this approach is that there is no need to define new algorithms for updates and repair, which have to be proven correct. Instead we will use a static compilation of the knowledge base according to the update or repair requirement. We prove that this transformation preserves the necessary semantics. A theorem prover can be used to compute the necessary update and repair actions. A hypertableau-based theorem prover like E-KRHyper is very well suited for this task, because the transformation enables it to calculate only the deviation of the ABox. Since E-KRHyper has recently been extended to deal with knowledge bases given in $\mathcal{SHIQ}$ [4], we chose to use this theorem prover.

Our approach is related to axiom pinpointing. For a given consequence, axiom pinpointing is the task to find the minimal subsets of the knowledge base under consideration, having this consequence. See [2] for details. In [12] laconic and precise justifications are introduced. Given an ontology and an entailment, a justification is a minimal subset of that ontology such that the entailment still holds in the subset. Roughly spoken, laconic justifications are not allowed to contain superfluous parts. In contrast to axiom pinpointing and justifications, we calculate subsets of the ABox and not of the whole knowledge base. In [20] inco-

herent TBoxes, i.e. TBoxes containing an unsatisfiable concept, are investigated. [11] considers so called syntactic ABbox updates. Similar to our approach, assertions are added to or removed from the ABox. In contrast to our approach, it is neither guaranteed that the removed assertion is not contained in the deductive closure nor that the result of adding the assertion is consistent.

In Section 2 we give both syntax and semantics of the Description Logic $\mathcal{SHI}$. In addition to that, we introduce the notion of DL-clauses as used in [17]. In Section 3 we give definitions for instance-level deletion, insertion and repair. Section 4 introduces the so called $\mathcal{K}^*$-transformation which in Section 5 is used to calculate the instance-level deletion, insertion and repair. The $\mathcal{K}^*$-transformation is implemented and in Section 6 we present first experimental results. Proofs of all theorems, propositions and lemmas can be found in [9].

2 $\mathcal{SHI}$ and DL-Clauses

First, we introduce the Description Logic $\mathcal{SHI}$. Given a set of *atomic roles* N_R, the set of *roles* is defined as $N_R \cup \{R^- \mid R \in N_R\}$, where R^- denotes the *inverse role* corresponding to the atomic role R. Let further Inv be a function on the set of roles that computes the inverse of a role, with $Inv(R) = R^-$ and $Inv(R^-) = R$. A *role inclusion axiom* is an expression of the form $R \sqsubseteq S$, where R and S are atomic or inverse roles. A *transitivity axiom* is of the form $Trans(S)$ for S an atomic or inverse role. An RBox $\mathcal{R}$ is a finite set of role inclusion axioms and transitivity axioms. $\sqsubseteq^*$ denotes the reflexive, transitive closure of $\sqsubseteq$ over $\{R \sqsubseteq S, Inv(R) \sqsubseteq Inv(S) \mid R \sqsubseteq S \in \mathcal{R}\}$. A role R is *transitive* in $\mathcal{R}$ if there exists a role S such that $S \sqsubseteq^* R$, $R \sqsubseteq^* S$, and either $Trans(S) \in \mathcal{R}$ or $Trans(Inv(S)) \in \mathcal{R}$. If no transitive role S with $S \sqsubseteq^* R$ exists, R is called *simple*.

Let N_C be the set of *atomic concepts*. The set of *concepts* is then defined as the smallest set containing $\top$, $\bot$, A, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$ and $\forall R.C$ for $A \in N_C$, C and D concepts and R a role.

A *general concept inclusion* (GCI) is of the form $C \sqsubseteq D$, and a TBox $\mathcal{T}$ is a finite set of GCIs.

Given a set of individuals N_I, an ABox $\mathcal{A}$ is a finite set of assertions of the form $A(a)$ and $R(a, b)$, with A an atomic concept, R an atomic role and a, b individuals from N_I. Note that in our setting, the ABox is only allowed to contain assertions about the belonging of individuals to atomic concepts and roles.

A knowledge base $\mathcal{K}$ is a triple $(\mathcal{R}, \mathcal{T}, \mathcal{A})$ with signature $\Sigma = (N_C, N_R, N_I)$. The tuple $\mathcal{I} = (\cdot^{\mathcal{I}}, \Delta^{\mathcal{I}})$ is an *interpretation* for $\mathcal{K}$ iff $\Delta^{\mathcal{I}}$ is a nonempty set and $\cdot^{\mathcal{I}}$ assigns an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ to each individual a, a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ to each atomic concept A, and a relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to each atomic role R. $\cdot^{\mathcal{I}}$ then assigns values to more complex concepts and roles as described in Table 1. $\mathcal{I}$ is a *model* of $\mathcal{K}$ ($\mathcal{I} \models \mathcal{K}$) if it satisfies all axioms and assertions in $\mathcal{R}$, $\mathcal{T}$ and $\mathcal{A}$ as shown in Table 1. A TBox $\mathcal{T}$ is called consistent, if there is an interpretation

satisfying all axioms in $\mathcal{T}$. A concept C is called *satisfiable w.r.t.* $\mathcal{R}$ and $\mathcal{T}$ iff there exists a model $\mathcal{I}$ of $\mathcal{R}$ and $\mathcal{T}$ with $C^{\mathcal{I}} \neq \emptyset$.

Concepts and Roles	
$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$	$(R^-)^{\mathcal{I}} = \{(y,x) \mid (x,y) \in R^{\mathcal{I}}\}$
$\bot^{\mathcal{I}} = \emptyset$	$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y : (x,y) \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}$
$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$	$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y : (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$	
$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$	
TBox & RBox axioms	**ABox axioms**
$C \sqsubseteq D \Rightarrow C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$	$C(a) \Rightarrow a^{\mathcal{I}} \in C^{\mathcal{I}}$
$R \sqsubseteq S \Rightarrow R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$	$R(a,b) \Rightarrow (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
$\mathit{Trans}(R) \Rightarrow (R^{\mathcal{I}})^+ \subseteq R^{\mathcal{I}}$	

Table 1. Model-theoretic semantics of $\mathcal{SHI}$. R^+ is the transitive closure of R.

In the sequel we adapt the notion of DL-clauses introduced in [17] to the Description Logic $\mathcal{SHI}$. These DL-clauses allow to use existent theorem provers which are based on the hypertableau calculus to compute models or to decide satisfiability. DL-clauses are universally quantified implications of the form $\bigvee V_j \leftarrow \bigwedge U_i$:

Definition 1. *([17]) An atom is of the form $B(s)$, $R(s,t)$, $\exists R.B(s)$ or $\exists R.\neg B(s)$ for B an atomic concept and s and t individuals or variables. An atom not containing any variables is called a ground atom. A DL-clause is of the form $V_1 \vee \ldots \vee V_n \leftarrow U_1 \wedge \ldots \wedge U_m$ with V_i atoms and U_j atoms of the form $B(s)$ or $R(s,t)$ and $m \geq 0$ and $n \geq 0$. If $n = 0$, we denote the left hand side (head) of the DL-clause by $\bot$. If $m = 0$, we denote the right hand side (body) of the DL-clause by $\top$.*

Definition 2. *(Semantics of DL-clauses; [17]) Let $V_1 \vee \ldots \vee V_n \leftarrow U_1 \wedge \ldots \wedge U_m$ be a DL-clause and N_V a set of variables, disjoint from N_I. Let further $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation and $\mu : N_V \rightarrow \Delta^{\mathcal{I}}$ be a variable mapping. Let $a^{\mathcal{I},\mu} = a^{\mathcal{I}}$ for an individual a and $x^{\mathcal{I},\mu} = \mu(x)$ for a variable x. Satisfaction of an atom, a DL-clause, and set of DL-clauses N in $\mathcal{I}$ and μ is defined as follows:*

$\mathcal{I}, \mu \models C(s)$ *if* $s^{\mathcal{I},\mu} \in C^{\mathcal{I}}$

$\mathcal{I}, \mu \models R(s,t)$ *if* $\langle s^{\mathcal{I},\mu}, t^{\mathcal{I},\mu} \rangle \in R^{\mathcal{I}}$

$\mathcal{I}, \mu \models \bigvee_{j=1}^{n} V_j \leftarrow \bigwedge_{i=1}^{m} U_i$ *if* $\mathcal{I}, \mu \models V_j$ *for some* $1 \leq j \leq n$ *whenever* $\mathcal{I}, \mu \models U_i$ *for each* $1 \leq i \leq m$

$\mathcal{I} \models \bigvee_{j=1}^{n} V_j \leftarrow \bigwedge_{i=1}^{m} U_i$ *if* $\mathcal{I}, \mu \models \bigvee_{j=1}^{n} V_j \leftarrow \bigwedge_{i=1}^{m} U_i$ *for all mappings* μ

$\mathcal{I} \models N$ *if* $\mathcal{I} \models r$ *for each DL-clause* $r \in N$

We will not give the transformation into DL-clauses. The details can be found in [17]. The transformation avoids an exponential blowup by using the

well-known structural transformation [18] and can be computed in polynomial time.

By $\Xi(\mathcal{T})$ ($\Xi(\mathcal{A})$) we denote the set of DL-clauses for a TBox $\mathcal{T}$ (an ABox $\mathcal{A}$). For a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, $\Xi(\mathcal{K}) = \Xi(\mathcal{T}) \cup \Xi(\mathcal{A})$. According to [17] for every interpretation $\mathcal{I}$, $\mathcal{I} \models \mathcal{K}$ iff $\mathcal{I} \models \Xi(\mathcal{T})$ and $\mathcal{I} \models \mathcal{A}$.

Since we assume the ABox assertions to be atomic, the ABox itself corresponds to a set of DL-clauses.

Example 1. The TBox $\mathcal{T} = \{B \sqsubseteq \exists R.C, \exists R.C \sqsubseteq D, D \sqsubseteq C\}$ corresponds to the set of DL-clauses $\Xi(\mathcal{T}) = \{\exists R.C(x) \leftarrow B(x), D(x) \leftarrow R(x,y) \wedge C(y), C(x) \leftarrow D(x)\}$.

Sometimes it is convenient to regard both the body and the head of a DL-clause C as a set of atoms like $C = \mathbf{H} \leftarrow \mathbf{B}$. This allows us to write $A \in \mathbf{B}$ ($A \in \mathbf{H}$) if atom A occurs in the body (in the head) of DL-clause C. The signature of a set of DL-clauses is the set of atomic concepts and atomic roles occurring in the DL-clause. The size of a DL-clause C is defined as the numbers of atoms occurring in C and is denoted $\mathit{size}(C)$. The size of a set of DL-clauses N denoted by $\mathit{size}(N)$ is the sum of sizes of all DL-clauses in N. In the sequel we need a function extracting the concept/role from an atom:

Definition 3. *(Symbol Extraction Function) Let A be an atom. Then $\sigma(A)$ is defined as follows:*

$$\sigma(A) = \begin{cases} B & \text{if } A = B(s) \text{ for some atomic concept } B, \\ R & \text{if } A = R(s,t) \text{ for some atomic role } R, \\ \exists R.B & \text{if } A = \exists R.B(s) \text{ for some atomic role } R \text{ and} \\ & B = E \text{ or } \neg E \text{ for some atomic concept } E. \end{cases}$$

By $\sigma(N)$ for a set of atoms N we denote the union of $\sigma(A)$ for all atoms $A \in N$.

In the following it is convenient for us to regard an interpretation as the set of ground atoms assigned to true by the interpretation. A set of ground atoms and an interpretation can be seen as equivalent, since every set of ground atoms uniquely determines a Herbrand interpretation. Now we can introduce the idea of minimal models to DL-clauses.

Definition 4. *(Minimal Model for a Set of DL-Clauses) Let DL be a set of DL-clauses. An Interpretation $\mathcal{I}$ is called a minimal model for DL, iff $\mathcal{I}$ is a model for DL and further there is no model $\mathcal{I}'$ for DL such that $\mathcal{I}' \subset \mathcal{I}$.*

Next we define minimality of models w.r.t. a set of ground atoms. We will later use this notion in order to minimize the number of ABox assertions which are to be deleted.

Definition 5. *(Γ Minimal Model) Let DL be a set of DL-clauses and Γ be a set of ground atoms. An interpretation $\mathcal{I}$ is a Γ-minimal model for DL iff $\mathcal{I}$ is a model for DL and further there is no model $\mathcal{I}'$ for DL with $\mathcal{I}' \cap \Gamma \subset \mathcal{I} \cap \Gamma$.*

3 ABox Evolution

We address three different operations on the instance level of the knowledge base: deletion, insertion and repair. The first scenario we are considering is the following: given a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ with a consistent TBox $\mathcal{T}$, we want to remove an ABox assertion for example $A(a)$ from the ABox. In general it is not sufficient to only delete $A(a)$ from the ABox, because $A(a)$ can still be contained in the deductive closure. So the task is to determine a minimal set of ABox assertions, which have to be deleted from the ABox in order to prevent that $A(a)$ is a logical consequence of the knowledge base. This leads to the following definition.

Definition 6. *(Minimal Instance Deletion) Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base where $\mathcal{T}$ is consistent. A ground atom D of the form $A(a)$ or $R(a, b)$ with $D \in \mathcal{A}$ is called delete request. Further $\mathcal{A}' \subseteq \mathcal{A}$ is called minimal instance deletion of D from $\mathcal{A}$ if $\mathcal{T} \cup \mathcal{A}' \not\models D$ and there is no $\mathcal{A}''$ with $\mathcal{A}' \subset \mathcal{A}'' \subseteq \mathcal{A}$ and $\mathcal{T} \cup \mathcal{A}'' \not\models D$.*

Example 2. We consider the ABox

$$\mathcal{A} = \{B(a), D(a), C(b), R(b, b), R(a, a)\}$$

together with the TBox given in Example 1. The delete request $D(a)$ has a minimal instance deletion

$$\mathcal{A}' = \{C(b), R(b, b), R(a, a)\}$$

Next we want to repair an ABox which is not consistent w.r.t. its TBox.

Definition 7. *(Minimal ABox Repair) Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base where $\mathcal{T}$ is consistent. $\mathcal{A}' \subseteq \mathcal{A}$ is called minimal ABox repair of $\mathcal{A}$ if $\mathcal{T} \cup \mathcal{A}'$ is consistent and there is no $\mathcal{A}''$ with $\mathcal{A}' \subset \mathcal{A}'' \subseteq \mathcal{A}$ and $\mathcal{T} \cup \mathcal{A}''$ consistent.*

Note that we define the notion of a minimal ABox repair in a way, that it is also applicable to an ABox which is consistent to its TBox. In this case, the minimal ABox repair corresponds to the original ABox.

The third instance level operation we address is insertion of an assertion into an existing ABox. The problem that arises when considering insertion is, that the resulting ABox might be inconsistent w.r.t. its TBox.

Example 3. Let us consider the set of DL-clauses

$$\Xi(\mathcal{T}) = \{\bot \leftarrow C(x) \wedge D(x)\}$$

together with the ABox

$$\mathcal{A} = \{C(a)\}$$

Adding the assertion $D(a)$ into $\mathcal{A}$ leads to $\mathcal{A}' = \{C(a), D(a)\}$, which is inconsistent w.r.t. $\mathcal{T}$.

We avoid inconsistent results by the next definition.

Definition 8. *(Minimal Instance Insertion) Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base with $\mathcal{T}$ consistent and D a ground atom of the form $A(a)$ or $R(a, b)$. An ABox $\mathcal{A}'$ is called minimal instance insertion of D into $\mathcal{A}$ if*

- $D \in \mathcal{A}'$,
- $(\mathcal{A}' \setminus D) \subseteq \mathcal{A}$,
- $\mathcal{T} \cup \mathcal{A}'$ *is consistent and there is no* $\mathcal{A}''$ *with* $D \in \mathcal{A}''$ *and* $(\mathcal{A}' \setminus D) \subset (\mathcal{A}'' \setminus D) \subseteq \mathcal{A}$ *and* $\mathcal{T} \cup \mathcal{A}''$ *is consistent.*

4 $\mathcal{K}^*$-Transformation

We will solve the tasks defined in Section 3 by using the $\mathcal{K}^*$-transformation which will be introduced in this section. As discussed in the introduction we want to use the ABox of the knowledge base as a partial model, which will guide our transformation.

Considering the task of deleting a given instance, we want to determine a minimal set of ABox assertions which have to be deleted in order to prevent the instance from being contained in the deductive closure of the knowledge base. The idea of the transformation we are about to use was introduced in [3]. We replace occurrences of an atom $A(a)$ in a clause by $\neg NegA(a)$. This transformation can be seen as switching the sides in the clause representation of DL-clauses. This makes sense, when a bottom-up proof procedure like E-KRHyper is used: a fact $A(a) \leftarrow$ changes the side and the clause becomes $\leftarrow NegA(a)$. As a consequence $A(a)$ is not derived explicitly. It is assumed to be in the model until the opposite has to be derived.

Deducing an atom $NegA(a)$ means that we have to revise the ABox and that we have to remove atom $A(a)$ from the ABox. By using this transformation we only need to calculate the atoms we have to remove from the ABox. All remaining atoms will be kept in the ABox. Since it is reasonable to expect the ABox to be very large, it is advantageous to calculate only the deviation from the original ABox.

Definition 9. *The Neg and the ABox function map atoms to renamed atoms:*

- *For atomic concepts A and an individual or variable a:*
  - $Neg(A(a)) = NegA(a)$
  - $ABox(A(a)) = ABoxA(a)$
- *For atomic roles R and individuals or variables a, b:*
  - $Neg(R(a, b)) = NegR(a, b)$
  - $ABox(R(a, b)) = ABoxR(a, b)$

We slightly abuse notation by using the *Neg* function to rename atomic concepts and atomic roles: for B an atomic concept or an atomic role: $Neg(B) = NegB$. Further for a set of atoms P, $Neg(P)$ is defined as: $Neg(P) = \{Neg(A) \mid A \in P\}$. So we can use the *Neg* function to rename atoms, sets of atoms and atomic concepts and roles.

Definition 10. [1] *(Renaming) Let DL be a set of DL-clauses and S a set of atomic concepts and atomic roles. Let $C \in DL$ be $C = \mathbf{H} \leftarrow \mathbf{B}$. Then $\mathsf{R}_S(C)$, the* renaming of *C w.r.t. S is*

$$\mathsf{R}_S(C) =$$

$$\{C\} \tag{1}$$

$$\cup$$

$$\{(\bigvee_{\substack{A\in\mathbf{H},\\ \sigma(A)\notin S}} A) \vee (\bigvee_{\substack{B\in\mathbf{B},\\ \sigma(B)\in S}} Neg(B)) \leftarrow (\bigwedge_{\substack{B\in\mathbf{B},\\ \sigma(B)\notin S}} B) \wedge (\bigwedge_{\substack{A\in\mathbf{H},\\ \sigma(A)\in S}} Neg(A))\} \tag{2}$$

$$\cup$$

$$\{\bot \leftarrow R(x,y) \wedge NegR(x,y) \mid \exists A \in (\mathbf{H} \cup \mathbf{B}) \text{ with } \sigma(A) = R \in S \text{ or } \exists A \in \mathbf{H} \text{ of the form } A = \exists R.C(z) \text{ and } R \in S\} \tag{3}$$

$$\cup$$

$$\{\bot \leftarrow D(x) \wedge NegD(x) \mid \exists A \in (\mathbf{H} \cup \mathbf{B}) \text{ with } \sigma(A) = D \in S \text{ or } \exists A \in \mathbf{H} \text{ of the form } A = \exists R.D(z) \text{ and } R \in S\} \tag{4}$$

For a set of DL-clauses DL, the renaming $\mathsf{R}_S(DL)$ w.r.t. S is defined as the union of the renaming of all its clauses.

Note that renaming is a bijective function on a set of DL-clauses. Further renaming can be performed in time linear to the size of the set of DL-clauses times the size of S.

The next proposition states the fact, that renaming preserves satisfiability. Furthermore given a model for a set of DL-clauses DL, it is possible to calculate a model for the renamed set of DL-clauses $\mathsf{R}_S(DL)$ and vice versa.

Proposition 1. *(Renaming Models) Let DL be a set of DL-clauses, S a set of atomic concepts and atomic roles and $\mathcal{I}$ an interpretation. Then $\mathcal{I} \models DL$ iff $\mathcal{I}^S \models \mathsf{R}_S(DL)$, where $\mathcal{I}^S$ and $\mathcal{I}$ have the same domain and the same interpretation of individuals. In addition to that the interpretation of all roles and concepts occurring in DL coincide. Further $(Neg(B))^{\mathcal{I}^S} = \overline{B^{\mathcal{I}}}$ for all concepts names $B \in S$ and $(Neg(R))^{\mathcal{I}^S} = \overline{R^{\mathcal{I}}}$ for all atomic roles $R \in S$.*

Definition 11. *($\mathcal{K}^*$-Transformation) Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base (where $\mathcal{T}$ is consistent). Let S be the set of atomic concepts and atomic roles occurring in $\mathcal{A}$. Then $\mathcal{K}^*$ is the clause set obtained by renaming $\Xi(\mathcal{T})$ w.r.t. S and adding the set of DL-clauses $\{ABox(A) \leftarrow \top \mid \text{for all assertions } A \in \mathcal{A}\}$.*

We have to add $\{ABox(A) \leftarrow \top \mid \text{for all assertions } A \in \mathcal{A}\}$ to the result of renaming for two reasons: first of all we have to introduce the individuals occurring in the ABox to the theorem prover. Furthermore it is helpful to calculate minimal deletions.

[1] Due to the helpful remarks of an anonymous reviewer of the DL Workshop, this definition was revised. These changes also affect the results presented in the experiments.

Proposition 2. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base and S be the set of atomic concepts and atomic roles occurring in $\mathcal{A}$ and $\mathcal{T}$. Then $\Xi(\mathcal{T})$, $R_S(\Xi(\mathcal{T}))$ and $\mathcal{K}^*$ are equisatisfiable.*

Example 4. We consider the set of DL-clauses given in Example 1 together with the ABox $\mathcal{A} = \{B(a), D(a), C(b), R(b,b), R(a,a)\}$. Then $S = \{B, D, C, R\}$. Renaming the DL-clauses given in Example 1 w.r.t. S leads to $\mathcal{K}^*$:

$$
\begin{aligned}
\exists R.C(x) &\leftarrow B(x).\\
\exists R.C(x) \vee \mathit{Neg}B(x) &\leftarrow \top.\\
D(x) &\leftarrow R(x,y) \wedge C(y).\\
\mathit{Neg}R(x,y) \vee \mathit{Neg}C(y) &\leftarrow \mathit{Neg}D(x).\\
C(x) &\leftarrow D(x).\\
\mathit{Neg}D(x) &\leftarrow \mathit{Neg}C(x).\\
\bot &\leftarrow R(x,y) \wedge \mathit{Neg}R(x,y).\\
\bot &\leftarrow C(x) \wedge \mathit{Neg}C(x).\\
\bot &\leftarrow B(x) \wedge \mathit{Neg}B(x).\\
\bot &\leftarrow D(x) \wedge \mathit{Neg}D(x).\\
\mathit{ABox}B(a) &\leftarrow \top.\\
\mathit{ABox}D(a) &\leftarrow \top.\\
\mathit{ABox}C(b) &\leftarrow \top.\\
\mathit{ABox}R(b,b) &\leftarrow \top.\\
\mathit{ABox}R(a,a) &\leftarrow \top.
\end{aligned}
$$

In the worst case the $\mathcal{K}^*$-transformation quadruples the size of a set of DL-clauses: S is the set of all concepts/roles occurring in the clause set. The ABox contains b assertions and the TBox consists of a single clause: $C = H_1 \vee \ldots \vee H_i \leftarrow B_1 \wedge \ldots \wedge B_j$ with $n = i + j$. W.l.o.g. the symbols of all atoms occurring in C are concepts. This set of DL-clauses has the size $n + b$. Renaming results in:

$$
\begin{aligned}
\{H_1 \vee \ldots \vee H_i &\leftarrow B_1 \wedge \ldots \wedge B_j,\\
\mathit{Neg}(B_1) \vee \ldots \vee \mathit{Neg}(B_j) &\leftarrow \mathit{Neg}(H_1) \wedge \ldots \wedge \mathit{Neg}(H_i),\\
\bot &\leftarrow \sigma(H_1)(x) \wedge \mathit{Neg}(\sigma(H_1))(x),\\
&\vdots\\
\bot &\leftarrow \sigma(H_i)(x) \wedge \mathit{Neg}(\sigma(H_i))(x),\\
\bot &\leftarrow \sigma(B_1)(x) \wedge \mathit{Neg}(\sigma(B_1))(x),\\
&\vdots\\
\bot &\leftarrow \sigma(B_j)(x) \wedge \mathit{Neg}(\sigma(B_j))(x),\\
\cup\{\mathit{ABox}(A) &\leftarrow \top \mid \text{for all assertions } A \in \mathcal{A}\}
\end{aligned}
$$

The first clause is the original clause from the TBox. Its size is n. The second clause is created by renaming and has size n. Then n clauses of size 2 follow. At the end of the clause set are b clauses of the form $ABox(A)$ each of size 1. All in all the resulting set of clauses has the size $n + n + 2 * n + b \leq 4 * (n + b)$, which is four times higher than the size of the original set of DL-clauses.

5 Using the $\mathcal{K}^*$-transformation for ABox Evolution

Firstly we address deletion: Recall that according to the definition of the *Neg* function, $Neg(\mathcal{A})$ is defined as $\{Neg(A) \mid A \in \mathcal{A}\}$. Next we show how to use $Neg(\mathcal{A})$-minimal models to calculate minimal instance deletions. For a given model M we construct $Del(M) = \{A \in \mathcal{A} \mid Neg(A) \in M\}$. Intuitively $Del(M)$ constitutes the set of ABox assertions supposed to be deleted from the ABox to obtain a minimal instance deletion.

Theorem 1. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base where $\mathcal{T}$ is consistent, S the set of atomic concepts and atomic roles occurring in $\mathcal{A}$, and D a delete request. Let M^S be a $Neg(\mathcal{A})$-minimal model for $\mathcal{K}^* \cup \{Neg(D) \leftarrow \top\}$. Then $\mathcal{A} \setminus Del(M^S)$ is a minimal instance deletion of D from $\mathcal{A}$.*

Proof by first showing $\mathcal{T} \cup (\mathcal{A} \setminus Del(M^S)) \not\models D$ by constructing a model according to Proposition 1 for $\mathcal{T} \cup (\mathcal{A} \setminus Del(M^S)) \cup \{\leftarrow D\}$ from M^S. And then showing that there is no $Del' \subset Del(M^S)$ with $\mathcal{T} \cup (\mathcal{A} \setminus Del') \not\models D$. See [9] for details.

Example 5. Now we delete $D(a)$ from the DL-clauses of our running example. For this, we add the clause $NegD(a) \leftarrow$ to the result of the $\mathcal{K}^*$ transformation given in Example 4. For lack of space we only give the relevant part of a $Neg(\mathcal{A})$ minimal model for this set of clauses:

$$M = \{ABoxB(a), ABoxD(a), ABoxC(b), ABoxR(b,b), ABoxR(a,a), \\ NegD(a), NegB(a), \ldots\}$$

This model gives us the minimal deletion: $\mathcal{A}' = \{C(b), R(b,b), R(a,a)\}$

Note that Theorem 1 can further be used for minimal deletion of a delete request D which belongs to the deductive closure of the knowledge base but is not contained in the ABox $\mathcal{A}$. (Meaning $D \notin \mathcal{A}$ but $\mathcal{T} \cup \mathcal{A} \models D$). In this case we only have to make sure, that $\sigma(D) \in S$. If $\sigma(D)$ does not occur in $\mathcal{A}$ we have to add D manually to S in order to render the instance deletion possible.

The $\mathcal{K}^*$-transformation introduced in Definition 11 can be used to repair an ABox, which is inconsistent w.r.t. its TBox. The basic idea is to replace each occurrence of $\bot$ in $\mathcal{T}$ by a new atom *false* and further add *false* to S. After that, we use the $\mathcal{K}^*$-transformation and construct the minimal instance deletion of *false* from the ABox. The resulting ABox is a minimal ABox repair.

Lemma 1. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base with consistent $\mathcal{T}$, $\mathcal{T}_{false}$ the TBox obtained from $\mathcal{T}$ by replacing every occurrence of $\bot$ by false, $\mathcal{A}_{false}$ be*

$\mathcal{A} \cup \{false\}$ *and* $\mathcal{K}_{false} = (\mathcal{T}_{false}, \mathcal{A}_{false})$*. Let* S *be the set of atomic concepts and roles occurring in* $\mathcal{A}$ *and* $\mathcal{T}$ *plus false. Then there is a* $Neg(\mathcal{A})$*-minimal model for* $\mathcal{K}^*_{false} \cup \{Negfalse \leftarrow \top\}$*.*

Corollary 1. *Let* $\mathcal{K} = (\mathcal{T}, \mathcal{A})$*,* $\mathcal{T}_{false}$ *and* S *be defined as in Lemma 1. Then* $\mathcal{A} \setminus Del(M)$ *is a minimal ABox repair for* $\mathcal{A}$ *for all* $Neg(\mathcal{A})$*-minimal models* M *for* $\mathcal{K}^*_{false} \cup \{Negfalse \leftarrow \top\}$*.*

Corollary 1 follows immediately from Theorem 1 with $D = false$. See [9] for both proofs. Lemma 1 together with Corollary 1 implies, that such a minimal ABox repair can always be constructed.

Next we consider a special case of deletion. For a given knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, Theorem 1 can only be used to construct a minimal instance deletion of D from $\mathcal{A}$ if $\mathcal{K}^* \cup \{Neg(D) \leftarrow \top\}$ is satisfiable. However if $\mathcal{K}^* \cup \{Neg(D) \leftarrow \top\}$ is not satisfiable, there is no $Neg(\mathcal{A})$-minimal model for $\mathcal{K}^* \cup \{Neg(D) \leftarrow \top\}$ and therefore we cannot use Theorem 1 for the construction of a minimal instance deletion.

Example 6. Let $\mathcal{T}$ be a TBox containing the assertion $\top \sqsubseteq C$ stating that everything belongs to the concept C. This corresponds to the DL-clause $C(x) \leftarrow \top$. Let us further consider the ABox: $\mathcal{A} = \{C(a), B(a), C(b), B(b)\}$ The $\mathcal{K}^*$-transformation leads to

$$\begin{aligned}\mathcal{K}^* = \{&C(x) \leftarrow \top,\\ &\bot \leftarrow NegC(x),\\ &\bot \leftarrow C(x) \wedge NegC(x),\\ &ABoxC(a),\\ &ABoxB(a),\\ &ABoxC(b),\\ &ABoxB(b)\}\end{aligned}$$

If we now want to delete $C(a)$ from $\mathcal{A}$, we have to construct $Neg(\mathcal{A})$-minimal models for $\mathcal{K}^* \cup \{NegC(a) \leftarrow \top\}$. However $\mathcal{K}^* \cup \{NegC(a) \leftarrow \top\}$ is unsatisfiable. So we are not able to construct a minimal instance deletion of $C(a)$ from $\mathcal{A}$ using Theorem 1. Taking a closer look at the TBox reveals the problem: the TBox claims, that everything has to belong to the concept C. So the only way to remove $C(a)$ from $\mathcal{A}$ is to remove individual a entirely from the ABox.

The next Theorem uses this idea and states how to construct minimal ABox deletions in the case that $\mathcal{K}^* \cup \{Neg(D) \leftarrow \top\}$ is unsatisfiable. Please note that the requirement of $\mathcal{T} \cup \mathcal{A}$ being consistent in the next theorem is not a limitation since we are always able to repair an ABox which is inconsistent with respect to its TBox using Corollary 1.

Theorem 2. *Let* $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ *be a knowledge base with* $\mathcal{T} \cup \mathcal{A}$ *consistent. Let further* S *be the set of atomic concepts and roles occurring in* $\mathcal{A}$ *and let* D *be a delete request with* $Ind(D)$ *the set of individuals occurring in* D*. If* $\mathcal{K}^* \cup$

$\{Neg(D) \leftarrow \top\}$ is unsatisfiable, then $\mathcal{A}' \subseteq \mathcal{A}$ is a minimal instance deletion of D from $\mathcal{A}$, where $\mathcal{A}'$ is obtained from $\mathcal{A}$ by removing all ABox assertions containing an individual from $Ind(D)$.

With the help of Theorem 1 and 2 we are now able to construct minimal instance-level deletions independent from the satisfiability of $\mathcal{K}^* \cup \{Neg(D) \leftarrow \top\}$.

Next we address the insertion of an assertion into an existing ABox. This can be obtained, by first adding the assertion to the ABox and afterwards constructing all possible minimal repairs for the resulting ABox. If the added assertion is not contained in any of these minimal ABox repairs, then it is not possible to insert the assertion into the ABox without rendering the ABox inconsistent w.r.t. its TBox. If there is a minimal repair containing the added assertion, then the insertion is possible and the respective minimal ABox repair gives us the result of the insertion.

Example 7. In the Example 3, we can repair $\mathcal{A}'$. There are two minimal ABox repairs for $\mathcal{A}'$: $\mathcal{A}'' = \{C(a)\}$ and $\mathcal{A}''' = \{D(a)\}$. The first minimal repair corresponds to deleting the previously inserted $D(a)$ and therefore is not desirable. The second minimal repair however allows us to keep the inserted assertion.

6 Experimental Results

We developed a prototypical implementation for deletion of ABox assertions using the $\mathcal{K}^*$-transformation. We use the E-KRHyper theorem prover to construct the $Neg(\mathcal{A})$-minimal models which lead us to the minimal deletions. Another theorem prover able to handle DL-clauses is HermiT [17]. However HermiT is not able to calculate $Neg(\mathcal{A})$-minimal models. This is why we chose the E-KRHyper theorem prover for our implementation. All tests were carried out on a computer featuring an AMD Phenom X6 1090T @ 3.2GHz and 8GB RAM. To the best of our knowledge, there is no system performing deletion of ABox assertions as described in this paper. This is why we cannot compare our system to another system.

In Section 4 we briefly discussed the complexity of the entire $\mathcal{K}^*$-transformation. There is a linear blow up of the knowledge base and there is also polynomial time complexity for performing the transformation. The real costs for performing the deletion, insertion and repair are caused by the theorem prover which has to compute the $Neg(\mathcal{A})$-minimal models. For an overview about this issue we refer to [8]. We use E-KRHyper for the construction of $Neg(\mathcal{A})$-minimal models. For this we extended E-KRHyper by a feature to construct Γ-minimal models in a bottom-up way. This extension renders it possible to give E-KRHyper a set of DL-clauses together with a set of predicate symbols P and an integer i. Then E-KRHyper only constructs models containing at most i instances of P predicates. During reasoning, E-KRHyper discards all models with more than i instances of P predicates. If E-KRHyper is not able to find a model with i or less instances of P predicates, it terminates by stating that the maximal number of

instances is reached. We use this feature to construct $Neg(\mathcal{A})$-minimal models: for S the set of concepts and roles occurring in the knowledge base, we first call E-KRHyper with $\mathcal{K}^*$, the set $Neg(S)$ and $i = 1$. We successively increase i until E-KRHyper either gives us a model or a proof for the unsatisfiability of the set of DL-clauses. This ensures that the first model given by E-KRHyper is a $Neg(\mathcal{A})$-minimal model.

We use the $\mathcal{ALCHI}$ ontology VICODI [2] for testing our approach. The smallest version of this ontology consists of 223 axioms in the TBox and RBox and 53653 ABox assertions. The larger versions of this ontology are generated by duplicating the assertions of the original ABox several times and changing the names of the individuals in the assertions. Unfortunately the repetitive structure of the larger versions of the ontology, resulting from this construction, is not suitable to test the efficiency of our approach. This is why we focus on the smallest version of the VICODI ontology. We construct different versions with increasing numbers of ABox assertions. The TBox and RBox remain unchanged. For each version of the so created ontologies we used 1000 different ABox assertions as a delete request D, calculated the $\mathcal{K}^*$-transformation and used E-KRHyper to calculate the minimal ABox deletion. In Figure 1 we show the results for the different ABox sizes we considered. For most of the delete request considered, it was sufficient to only remove the delete request itself from the ABox. We call those cases atomic deletions. If more than one ABox assertion has to be deleted, we speak of non-atomic deletions. Figure 1 gives information on the average time used for a delete request leading to an atomic deletion as well as leading to a non-atomic deletion. Another way to determine atomic deletions is to use E-KRHyper without the $\mathcal{K}^*$-transformation. If we want to test, if D can be removed from the ABox by deleting only D from the ontology KB, we can test $KB \setminus \{D\} \cup \{\neg D\}$ for satisfiability using E-KRHyper. Satisfiability of $KB \setminus \{D\} \cup \{\neg D\}$ implies, that $KB \setminus \{D\} \not\models D$. Meaning that D can be deleted atomically. Note that this test can only be used for atomic deletions and is completely useless for the calculation of non-atomic deletions. You can find the time used for those atomic deletions computed by E-KRHyper without the $\mathcal{K}^*$-Transformation in Figure 1. Comparing the lines for E-KRHyper and atomic deletions using the $\mathcal{K}^*$-transformation shows, that the $\mathcal{K}^*$-Transformation is faster in calculating atomic deletions. In addition to that the $\mathcal{K}^*$-Transformation is able to calculate non-atomic deletions as well and is therefore better suited for deletion than E-KRHyper. Figure 1 reveals another nice property of the $\mathcal{K}^*$-transformation: increasing the size of the ABox only leads to a harmless increase of the time necessary to calculate the minimal deletion. We owe this property to the fact, that we only calculate the deviation from the original ABox. For the calculation of non-atomic deletions more than one run of E-KRHyper is necessary. This explains why non-atomic deletions take longer than atomic deletions. However the time necessary to calculate a non-atomic deletion only increases moderately when the size of the ABox under consideration is increased.

[2] http://www.vicodi.org

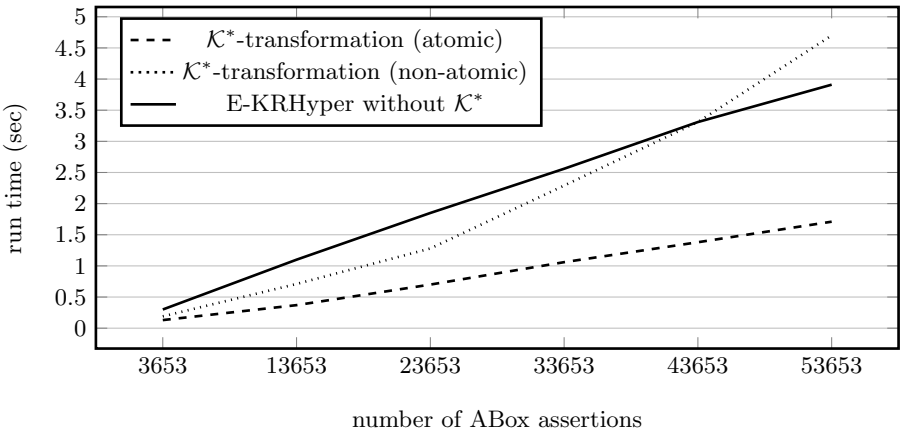

Figure 1. Time used for atomic and non-atomic deletions.

7 Conclusion and Future Work

In this paper we give a semantically guided compilation technique, the so called $\mathcal{K}^*$-transformation, for $\mathcal{SHI}$ knowledge bases. The transformed knowledge base is equisatisfiable to the original one. A theorem prover can be used for the computation of the necessary actions for deletion, insertion and repair from the result of the $\mathcal{K}^*$-transformation. Especially theorem provers based on a hypertableau calculus are suited for these computations. The approach is implemented and we introduced first experimental results using the theorem prover E-KRHyper.

In future work, we want to extend our implementation to enable it to do ABox repair and insertion of assertions as well.

Since E-KRHyper is able to handle the DL $\mathcal{SHIQ}$, we plan to extend our approach to qualified number restrictions.

References

1. Chandrabose Aravindan and Peter Baumgartner. Theorem proving techniques for view deletion in databases. *Journal of Symbolic Computation*, 29:2000, 2000.
2. Franz Baader and Rafael Peñaloza. Axiom pinpointing in general tableaux. *J. Log. Comput.*, 20(1):5–34, 2010.
3. Peter Baumgartner, Peter Fröhlich, Ulrich Furbach, and Wolfgang Nejdl. Semantically Guided Theorem Proving for Diagnosis Applications. In M. E. Pollack, editor, *IJCAI 97*, Nagoya, 1997. Morgan Kaufmann.
4. Markus Bender, Björn Pelzer, and Claudia Schon. System description: E-KRHyper 1.4 - extensions for unique names and description logic. In Maria Paola Bonacina, editor, *CADE-24*, LNCS, 2013.

5. Diego Calvanese, Evgeny Kharlamov, Werner Nutt, and Dmitriy Zheleznyakov. Updating aboxes in DL-Lite. In Alberto H. F. Laender and Laks V. S. Lakshmanan, editors, *AMW*, volume 619 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2010.
6. Heng Chu and David A. Plaisted. Semantically guided first-order theorem proving using hyper-linking. In Alan Bundy, editor, *CADE-12*, volume 814 of *LNCS*. Springer, 1994.
7. Giuseppe De Giacomo, Maurizio Lenzerini, Antonella Poggi, and Riccardo Rosati. On instance-level update and erasure in description logic ontologies. *J. Log. and Comput.*, 19, 2009.
8. Jürgen Dix, Ulrich Furbach, and Ilkka Niemelä. Nonmonotonic reasoning: Towards efficient calculi and implementations. In John Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, pages 1241–1354. Elsevier and MIT Press, 2001.
9. Ulrich Furbach and Claudia Schon. Semantically guided evolution of SHI aboxes. Reports of the Faculty of Informatics 4/2013, Universität Koblenz-Landau, 2013. Available at http://www.uni-koblenz.de/FB4/Publications/Reports.
10. Bernardo Cuenca Grau, Ernesto Jimenez Ruiz, Evgeny Kharlamov, and Dimitry Zhelenyakov. Ontology evolution under semantic constraints. In *Proc. of the 13th Int. Conference on Principles of Knowledge Representation and Reasoning*, 2012.
11. Christian Halashek-Wiener, Bijan Parsia, and Evren Sirin. Description logic reasoning with syntactic updates. In *Proc. of the 2006 Confederated international conference on On the Move to Meaningful Internet Systems: CoopIS, DOA, GADA, and ODBASE - Volume Part I*, ODBASE'06/OTM'06. Springer, 2006.
12. Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Laconic and precise justifications in owl. In *The Semantic Web-ISWC 2008*. Springer, 2008.
13. Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Explaining inconsistencies in owl ontologies. In Lluis Godo and Andrea Pugliese, editors, *SUM*, volume 5785 of *LNCS*. Springer, 2009.
14. Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant semantics for description logics. In Pascal Hitzler and Thomas Lukasiewicz, editors, *RR*, volume 6333 of *LNCS*. Springer, 2010.
15. Maurizio Lenzerini and Domenico Fabio Savo. On the evolution of the instance level of DL-Lite knowledge bases. In Riccardo Rosati, Sebastian Rudolph, and Michael Zakharyaschev, editors, *Description Logics*, volume 745 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.
16. Hongkai Liu, Carsten Lutz, Maja Milicic, and Frank Wolter. Foundations of instance level updates in expressive description logics. *Artificial Intelligence*, 175(18):2170–2197, 2011.
17. Boris Motik, Rob Shearer, and Ian Horrocks. Optimized Reasoning in Description Logics using Hypertableaux. In Frank Pfenning, editor, *CADE-21*, volume 4603 of *LNAI*. Springer, 2007.
18. David A. Plaisted and Steven Greenbaum. A structure-preserving clause form translation. *J. Symb. Comput.*, 2(3):293–304, 1986.
19. Riccardo Rosati. On the complexity of dealing with inconsistency in description logic ontologies. In *IJCAI'11*. AAAI Press, 2011.
20. Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In Georg Gottlob and Toby Walsh, editors, *IJCAI*. Morgan Kaufmann, 2003.