6-1-2007

Algorithms for Paraconsistent Reasoning with OWL

Yue Ma

Pascal Hitzler
pascal.hitzler@wright.edu

Zuoquan Lin

Algorithms for Paraconsistent Reasoning with OWL *

Yue Ma[1,2], Pascal Hitzler[2], and Zuoquan Lin[1]

[1]Department of Information Science, Peking University, China
[2]AIFB, Universität Karlsruhe, Germany
{mayue,lz}@is.pku.edu.cn, {yum,hitzler}@aifb.uni-karlsruhe.de

Abstract. In an open, constantly changing and collaborative environment like the forthcoming Semantic Web, it is reasonable to expect that knowledge sources will contain noise and inaccuracies. Practical reasoning techniques for ontologies therefore will have to be tolerant to this kind of data, including the ability to handle inconsistencies in a meaningful way. For this purpose, we employ paraconsistent reasoning based on four-valued logic, which is a classical method for dealing with inconsistencies in knowledge bases. Its transfer to OWL DL, however, necessitates the making of fundamental design choices in dealing with class inclusion, which has resulted in differing proposals for paraconsistent description logics in the literature. In this paper, we build on one of the more general approaches which due to its flexibility appears to be most promising for further investigations. We present two algorithms suitable for implementation, one based on a preprocessing before invoking a classical OWL reasoner, the other based on a modification of the KAON2 transformation algorithms. We also report on our implementation, called ParOWL.

1 Introduction

Real knowledge bases and data for Semantic Web applications will rarely be perfect. They will be distributed and multi-authored. They will be assembled from different sources and reused. It is unreasonable to expect such realistic knowledge bases to be always logically consistent, and it is therefore important to study ways of dealing with inconsistent knowledge. This is particularly important if the full power of logic-based approaches like the Web Ontology Language OWL [1] shall be employed, as classical logic breaks down in the presence of inconsistent knowledge.

The study of inconsistency handling in Artificial Intelligence has a long tradition, and corresponding results are recently being transferred to description logics, which underly OWL. Two fundamentally different approaches can be distinguished. The first is based on the assumption that inconsistencies indicate erroneous data which is to be repaired in order to obtain a consistent knowledge base, e.g. by selecting consistent

* We acknowledge support by the German Federal Ministry of Education and Research (BMBF) under the SmartWeb project (grant 01 IMD01 B), by the EU under the IST project NeOn (IST-2006-027595, http://www.neon-project.org/), the IST-FP6-026978 X-Media project, by the Deutsche Forschungsgemeinschaft (DFG) in the ReaSem project, and by China Scholarship Council, and partially by NSFC (grant numbers 60373002 and 60496322) and by NKBRPC (2004CB318000).

subsets for the reasoning process [2, 3]. The other approach yields to the insight that inconsistencies are a natural phenomenon in realistic data which are to be handled by a logic which tolerates it [4–6]. Such logics are called paraconsistent, and the most prominent of them are based on the use of additional truth values standing for *undefined* (i.e. neither true nor false) and *overdefined* (or *contradictory*, i.e. both true and false). Such logics are appropriately called *four-valued logics* [7]. We believe that either of the approaches is useful, depending on the application scenario.

In this paper, we contribute to the paraconsistency approach. We indeed extend on the preliminary work in [6], which has the following features.

- It is grounded in prominent research results from Artificial Intelligence [8].
- It is very flexible in terms of design choices which can be made when developing a paraconsistent description logic. This concerns the issues arising from the fact that there are different ways of defining the notion of logical implication in four-valued logics. The approach which we follow allows the full and simultaneous use of the different notions of implication.
- It does not increase worst-case computational complexity of reasoning if compared to standard reasoning methods for consistent knowledge bases.

In this paper, we present two algorithms for practical paraconsistent reasoning based on this approach. The first one is based on a transformation from a paraconsistent ontology O to a classical two-valued ontology $\overline{O}$ in such a way that paraconsistent reasoning on O can be simulated by classical reasoning on $\overline{O}$. The second algorithm is based on an adaptation of the algorithms underlying the KAON2 OWL Reasoner[1] [9] by realizing a resolution-based decision procedure for paraconsistent reasoning. We spell out the details for the description logic $\mathcal{ALC}$, which is considered to be the most foundational one and comprises a large fragment of OWL DL.

The paper is structured as follows. We first review briefly preliminaries in Section 2. In Section 3 we then describe the syntax and semantics of the paraconsistent description logic which we will use. Sections 4 and 5 describe the two reasoning procedures, respectively based on a transformation for preprocessing and on an adaptation of the KAON2 algorithms. In Section 6, we describe the prototype of our paraconsistent approach to reasoning with a possible inconsistent ontology, and discuss future work and conclude in Section 7.

This paper is a substantial continuation of work presented as preliminary results in [6]. Due to space limitations, proofs are omitted. They can be found in a technical report available from http://www.aifb.uni-karlsruhe.de/WBS/phi/pub/parowltr.pdf.

2 Preliminaries

2.1 The Description Logic $\mathcal{ALC}$

We briefly review notation and terminology of the description logic $\mathcal{ALC}$, but we basically assume that the reader is familiar with description logics. For comprehensive background reading, please refer to [10].

[1] http://kaon2.semanticweb.org

Table 1. Syntax and semantics of $\mathcal{ALC}$

Constructor Name	**Syntax**	**Semantics**
atomic concept A	A	$A^I \subseteq \Delta^I$
abstract role R_A	R	$R^I \subseteq \Delta^I \times \Delta^I$
individuals I	o	$o^I \in \Delta^I$
top concept	$\top$	Δ^I
bottom concept	$\bot$	$\emptyset$
conjunction	$C_1 \sqcap C_2$	$C^I \cap D^I$
disjunction	$C_1 \sqcup C_2$	$C^I \cup D^I$
negation	$\neg C$	$\Delta^I \setminus C^I$
exists restriction	$\exists R.C$	$\{x \mid \exists y, (x,y) \in R^I \text{ and } y \in C^I\}$
value restriction	$\forall R.C$	$\{x \mid \forall y, (x,y) \in R^I \text{ implies } y \in C^I\}$

Axiom Name	**Syntax**	**Semantics**
concept inclusion	$C_1 \sqsubseteq C_2$	$C_1^I \subseteq C_2^I$
concept assertion	$C(a)$	$a^I \in C^I$
role assertion	$R(a,b)$	$(a^I, b^I) \in R^I$

We assume that we are given a set of atomic concepts (or concept names), a set of roles (or role names), and a set of individuals. With the symbols $\top$ and $\bot$ we furthermore denote the top concept and the bottom concept, respectively.

Complex concepts in $\mathcal{ALC}$ can be formed from these inductively as follows.

1. $\top$, $\bot$, and each atomic concept are concepts;
2. If C, D are concepts, then $C \sqcup D$, $C \sqcap D$, and $\neg C$ are concepts;
3. If C is a concept and R is a role, then $\forall R.C$ and $\exists R.C$ are concepts.

An $\mathcal{ALC}$ ontology consists of a set of assertions, called the $ABox$ of the ontology, and a set of inclusion axioms, calld the $TBox$ of the ontology. Assertions are of the form $C(a)$ or $R(a,b)$, where a, b are individuals and C and R are concepts and roles, respectively. Inclusion axioms are of the form $C \sqsubseteq D$, where C and D are concepts. Informally, an assertion $C(a)$ means that the individual a is an instance of concept C, and an assertion $R(a,b)$ means that individual a is related with individual b via the property R. The inclusion axiom $C \sqsubseteq D$ means that each individual of C is an individual of D.

The formal definition of the (model-theoretic) semantics of $\mathcal{ALC}$ is given by means of interpretations $I = (\Delta^I, \cdot^I)$ consisting of a non-empty domain Δ^I and a mapping $\cdot^I$ satisfying the conditions in Table 1, interpreting concepts as subsets of the domain and roles as binary relations on the domain. An interpretation satisfies an $\mathcal{ALC}$ ontology (i.e. is a model of the ontology) iff it satisfies each axiom in both the $ABox$ and the $TBox$. An ontology is called satisfiable (unsatisfiable) iff there exists (does not exist) such a model. In $\mathcal{ALC}$, reasoning tasks, i.e. the derivation of logical consequences, can be reduced to satisfiability checking of ontologies [10, 11].

Table 2. Truth table for 4-valued connectives

α	f	f	f	f	t	t	t	t	$\ddot{\top}$	$\ddot{\top}$	$\ddot{\top}$	$\ddot{\top}$	$\ddot{\bot}$	$\ddot{\bot}$	$\ddot{\bot}$	$\ddot{\bot}$
β	f	t	$\ddot{\top}$	$\ddot{\bot}$	f	t	$\ddot{\top}$	$\ddot{\bot}$	f	t	$\ddot{\top}$	$\ddot{\bot}$	f	t	$\ddot{\top}$	$\ddot{\bot}$
$\neg\alpha$	t	t	t	t	f	f	f	f	$\ddot{\top}$	$\ddot{\top}$	$\ddot{\top}$	$\ddot{\top}$	$\ddot{\bot}$	$\ddot{\bot}$	$\ddot{\bot}$	$\ddot{\bot}$
$\alpha \wedge \beta$	f	f	f	f	f	t	$\ddot{\top}$	$\ddot{\bot}$	f	$\ddot{\top}$	$\ddot{\top}$	f	f	$\ddot{\bot}$	f	$\ddot{\bot}$
$\alpha \vee \beta$	f	t	$\ddot{\top}$	$\ddot{\bot}$	t	t	t	t	$\ddot{\top}$	t	$\ddot{\top}$	t	$\ddot{\bot}$	t	t	$\ddot{\bot}$
$\alpha \mapsto \beta$	t	t	t	t	f	t	$\ddot{\top}$	$\ddot{\bot}$	$\ddot{\top}$	t	$\ddot{\top}$	t	$\ddot{\bot}$	t	t	$\ddot{\bot}$
$\alpha \supset \beta$	t	t	t	t	f	t	$\ddot{\top}$	$\ddot{\bot}$	f	t	$\ddot{\top}$	$\ddot{\bot}$	t	t	t	t
$\alpha \rightarrow \beta$	t	t	t	t	f	t	f	$\ddot{\bot}$	f	t	$\ddot{\top}$	$\ddot{\bot}$	$\ddot{\bot}$	t	$\ddot{\bot}$	t

2.2 Four-valued Logic

The major studies of four-valued logics have been carried out in the setting of propositional logic. We will very briefly review the preliminaries which set the state for the four-valued version of $\mathcal{ALC}$ which we will present later in Section 3.

The idea of four-valued logic is based on the idea of having four truth values, instead of the classical two. The four truth values stand for *true, false, unknown* (or *undefined*) and *both* (or *overdefined, contradictory*). We use the symbols $t, f, \ddot{\bot}, \ddot{\top}$, respectively, for these truth values, and the set of these four truth values is denoted by FOUR. The truth value $\ddot{\top}$ stands for contradictory information, hence four-valued logic lends itself to dealing with inconsistent knowledge. The value $\ddot{\top}$ thus can be understood to stand for *true and false*, while $\ddot{\bot}$ stands for *neither true nor false*, i.e. for the absence of any information about truth or falsity.

Syntactically, four-valued logic is very similar to classical logic. Care has to be taken, however, in defining meaningful notions of implication, as there are several ways to do this. Indeed, there are three major notions of implication in the literature, all of which we will employ in our approach. The logical connectives we allow are thus negation $\neg$, disjunction $\vee$, conjunction $\wedge$, material implication $\mapsto$, internal implication $\supset$, and strong implication $\rightarrow$. We will discuss them in detail later on as the presence of all three implications is crucial for our approach.

Four-valued interpretations for formulae (i.e. 4-interpretations) are obviously mappings from formulae to (the set of four) truth values, respecting the truth tables for the logical connectives, as detailed in Table 2. Four-valued models (4-models) are defined in the obvious way, as follows, where t and $\ddot{\top}$ are the designated truth values.

Definition 1 *Let I be a 4-interpretation, let Σ be a theory (i.e. set of formulae) and let φ be a formula in four-valued logic. Then I is a 4-model of φ if and only if $I(\varphi) \in \{t, \ddot{\top}\}$. I is a 4-model of Σ if and only if I is a 4-model of each formula in Σ. Σ four-valued entails φ, written $\Sigma \models_4 \varphi$, if and only if every 4-model of Σ is a 4-model of φ.*

Proposition 1. *We note the following general properties.*

- *The language* $L = \{\neg, \vee, \wedge, \supset, \ddot{\bot}, \ddot{\top}\}$ *is functional complete for the set* FOUR *of truth values, i.e. every function from* FOUR^n *to* FOUR *is representable by some formula in L [8, Theorem 12].*
- *Any formula containing only connectives from* $\{\neg, \vee, \wedge, \supset\}$ *always has a four-valued model.*

Some general remarks about the different notions of implication are in order. They are the major notions of implication used in the literature, and are discussed in detail in [8, 12]. The basic rationales behind them are the following: Material implication can be defined by means of negation and disjunction as known from classical logic. However, it does not satisfy Modus Ponens or the deduction theorem, and is thus of limited use as an *implication* in the intuitive sense. Internal implication satisfies Modus Ponens and the deduction theorem, but cannot be defined by means of other connectives. Furthermore, internal implication does not satisfy contraposition. Strong implication is stronger than internal implication, in that it additionally satisfies contraposition. Indeed, an alternative view on the truth tables for the implication connectives is as follows.

$$\varphi \mapsto \psi \text{ is definable as } \neg\varphi \vee \psi. \qquad \text{(Material Implication)}$$

$$\varphi \supset \psi \quad \text{evaluates to} \quad \begin{cases} \psi & \text{if } \varphi \in \{t, \ddot{\top}\} \\ t & \text{if } \varphi \in \{f, \ddot{\bot}\} \end{cases} \qquad \text{(Internal Implication)}$$

$$\varphi \rightarrow \psi \text{ is definable as } (\varphi \supset \psi) \wedge (\neg\psi \supset \neg\varphi) \qquad \text{(Strong Implication)}$$

Further properties of the implication connectives are summarized in the following proposition (as shown in [8, Corollary 9] and [12]).

Proposition 1 *The following claims hold, where* Γ *is a theory and* ψ, ϕ *are formulae.*

- *Internal implication is not definable in terms of the connectives* $\neg, \vee, \wedge$.
- $\Gamma, \psi \models_4 \phi$ *iff* $\Gamma \models_4 \psi \supset \phi$.
- *If* $\Gamma \models_4 \psi$ *and* $\Gamma \models_4 \psi \supset \phi$ *then* $\Gamma \models_4 \phi$.
- $\psi \rightarrow \phi$ *implies that* $\neg\phi \rightarrow \neg\psi$.

Apart from the formal properties of the different notions of implication, it is obviously important to consider their intuitive meaning and their usefulness for knowledge base modelling. We will discuss this in detail in the next section.

3 The Four-valued Description Logic $\mathcal{ALC}4$

We describe the syntax and semantics of our four-valued description logic $\mathcal{ALC}4$. The approach is fairly standard apart from the fact that we allow the simultaneous use of all three notions of implication introduced in Section 2.2. We will thus devote significant space to a detailed discussion of the intuitions behind these different implications.

Syntactically, $\mathcal{ALC}4$ hardly differs from $\mathcal{ALC}$. Complex concepts and assertions are defined in exactly the same way. For class inclusion, however, the question arises how to interpret the underlying implication connective in the four-valued setting. We thus allow

Table 3. Semantics of $\mathcal{ALC}4$ Concepts

Constructor Syntax	Semantics
A	$A^I = \langle P, N\rangle$, where $P, N \subseteq \Delta^I$
R	$R^I = \langle R_P, R_N\rangle$, where $R_P, R_N \subseteq \Delta^I \times \Delta^I$
o	$o^I \in \Delta^I$
$\top$	$\langle \Delta^I, \emptyset\rangle$
$\bot$	$\langle \emptyset, \Delta^I\rangle$
$C_1 \sqcap C_2$	$\langle P_1 \cap P_2, N_1 \cup N_2\rangle$, if $C_i^I = \langle P_i, N_i\rangle$ for $i = 1, 2$
$C_1 \sqcup C_2$	$\langle P_1 \cup P_2, N_1 \cap N_2\rangle$, if $C_i^I = \langle P_i, N_i\rangle$ for $i = 1, 2$
$\neg C$	$(\neg C)^I = \langle N, P\rangle$, if $C^I = \langle P, N\rangle$
$\exists R.C$	$\langle \{x \mid \exists y, (x, y) \in \text{proj}^+(R^I)$ and $y \in \text{proj}^+(C^I)\}$, $\{x \mid \forall y, (x, y) \in \text{proj}^+(R^I)$ implies $y \in \text{proj}^-(C^I)\}\rangle$
$\forall R.C$	$\langle \{x \mid \forall y, (x, y) \in \text{proj}^+(R^I)$ implies $y \in \text{proj}^+(C^I)\}$, $\{x \mid \exists y, (x, y) \in \text{proj}^+(R^I)$ and $y \in \text{proj}^-(C^I)\}\rangle$

three kinds of class inclusions, corresponding to the three implication connectives we have discussed. They are as follows, $C \mapsto D$, $C \sqsubset D$, and $C \rightarrow D$, called material inclusion axiom, internal inclusion axiom, and strong inclusion axiom, respectively.

Semantically, interpretations map individuals to elements of the domain of the interpretation, as usual. For concepts, however, we need to make modifications to the notion of interpretation in order to allow for reasoning with inconsistencies.

Intuitively, in four-valued logic we need to consider four situations which can occur in terms of containment of an individual in a concept: (1) we know it is contained, (2) we know it is not contained, (3) we have no knowledge whether or not the individual is contained, (4) we have contradictory information, namely that the individual is both contained in the concept and not contained in the concept. There are several equivalent ways how this intuition can be formalised, one of which is described in the following.

For a given domain Δ^I and a concept C, an interpretation over Δ^I assigns to C a pair $\langle P, N\rangle$ of (not necessarily disjoint) subsets of Δ^I. Intuitively, P is the set of elements known to belong to the extension of C, while N is the set of elements known to be not contained in the extension of C. For simplicity of notation, we define functions $\text{proj}^+(\cdot)$ and $\text{proj}^-(\cdot)$ by $\text{proj}^+\langle P, N\rangle = P$ and $\text{proj}^-\langle P, N\rangle = N$.

Formally, a four-valued interpretation is a pair $I = (\Delta^I, \cdot^I)$ with Δ^I as domain, where $\cdot^I$ is a function assigning elements of Δ^I to individuals, and subsets of $(\Delta^I)^2$ to concepts, such that the conditions in Table 3 are satisfied. Note that the conditions in Table 3 for role restrictions are designed in such a way that the logical equivalences $\neg(\forall R.C) = \exists R.(\neg C)$ and $\neg(\exists R.C) = \forall R.(\neg C)$ are retained – this is the most convenient way for us for handling role restrictions, as it will allows for a straightforward translation from $\mathcal{ALC}4$ to classical $\mathcal{ALC}$. Note also that for roles we actually require only the positive part of the extension – we nevertheless require interpretations to assign pairs of sets to roles, which is a technical formality to retain consistency of notation with possible extensions to more expressive description logics (see [6]).

Obviously, under the constraints $P \cap N = \emptyset$ and $P \cup N = \Delta$, four-valued interpretations become just standard two-valued interpretations.

Table 4. Semantics of inclusion axioms in $\mathcal{ALC}4$

Axiom Name	Syntax	Semantics
material inclusion	$C_1 \mapsto C_2$	$\Delta^I \setminus \text{proj}^-(C_1^I) \subseteq \text{proj}^+(C_2^I)$
internal inclusion	$C_1 \sqsubset C_2$	$\text{proj}^+(C_1^I) \subseteq \text{proj}^+(C_2^I)$
strong inclusion	$C_1 \rightarrow C_2$	$\text{proj}^+(C_1^I) \subseteq \text{proj}^+(C_2^I)$ and $\text{proj}^-(C_2^I) \subseteq \text{proj}^-(C_1^I)$
concept assertion	$C(a)$	$a^I \in \text{proj}^+(C^I)$
role assertion	$R(a,b)$	$(a^I, b^I) \in \text{proj}^+(R^I)$

The correspondence between truth values from FOUR and concept assertions is the obvious one: For instances $a \in \Delta^I$ and concept name C we have

- $C^I(a) = t(\ddot{\top})$, iff $a^I \in \text{proj}^+(C^I)$ and $a^I \notin (\in)\text{proj}^-(C^I)$,
- $C^I(a) = f(\ddot{\bot})$, iff $a^I \notin \text{proj}^+(C^I)$ and $a^I \in (\notin)\text{proj}^-(C^I)$,

When defining the semantics as we just did, we ensure that a number of useful equivalences from classical logic hold, as follows.

Proposition 2 *For any four-valued interpretation I and concepts C, D, the following claims hold.*

$$(C \sqcap \top)^I = C^I, (C \sqcup \top)^I = \top^I, (C \sqcap \bot)^I = \bot^I, (C \sqcup \bot)^I = C^I,$$
$$(\neg\neg C)^I = C^I, (\neg\top)^I = \bot^I, (\neg\bot)^I = \top^I, (\neg(C \sqcup D))^I = (\neg C \sqcap \neg D)^I,$$
$$(\neg(C \sqcap D))^I = (\neg C \sqcup \neg D)^I, (\neg(\forall R.C))^I = (\exists R.\neg C)^I, (\neg(\exists R.C))^I = (\forall R.\neg C)^I.$$

We now come to the semantics of the three different types of inclusion axioms. It is formally defined in Table 4 (together with the semantics of concept assertions). We say that a four-valued interpretation I satisfies a four-valued ontology $\mathcal{O}$ (i.e. is a model of it) iff it satisfies each assertion and each inclusion axiom in $\mathcal{O}$. An ontology $\mathcal{O}$ is 4-valued satisfiable (unsatisfiable) iff there exists (does not exist) such a model.

With the formal definitions out of the way, it remains to address the intuitions underlying the different inclusion axioms. These intuitions are evidenced by the formal properties of the underlying implications as discussed in Section 2.2 as well as the behaviour of the implications in practice. We actually foresee a possible workflow for handling inconsistent ontologies, as follows. In a first step, inclusion axioms are classified into the three types of four-valued inclusion axioms available. Then four-valued reasoning is performed based on the classification, in order to arrive at a meaningful 4-valued conclusion. The question, how such a classification can be performed, will not be addressed in this paper. It constitutes a seperate substantial piece of work which is under investigation by the authors. A combination of automated detection and a user-interaction process may be the most workable solution, where the user-interaction process may be guided by the intuitive explanations which we will now give for the three types of inclusion. An example which displays the effects of the different inclusions is given in Section 6.

Strong inclusion respects the deduction theorem and contraposition reasoning. In a paraconsistent context, it is thus the inclusion to be used for universal truth, such as Square $\rightarrow$ FourEdged.

Internal inclusion propagates contradictory information forward, but not backward as it does not allow for contraposition reasoning. It can thus be characterized as a *brave* way of handling inconsistency. It should be used whenever it is important to infer the consequent even if the antecedent may be contradictory. To give an example, consider a robot fault diagnosis system and an axiom stating that oil leakage is indicative of a robot malfunction. Obviously, it is important to check on a possible malfunction even in case there is contradictory information about an oil leakage. In a paraconsistent context, the axiom is thus best modeled by means of internal inclusion, i.e. as OilLeakage $\sqsubset$ RobotMalfunction.

Material inclusion is *cautious* in the sense that contradictory information is not propagated. The intuition behind material inclusion becomes apparent by studying the truth table for material implication: $a \mapsto b$ indicates that the only way for b to be *not true* (i.e. to be f or $\ddot{\bot}$) is if there is information of falsity of a (i.e. it is f or $\ddot{\top}$). This kind of modeling becomes important if an inclusion has to be second-guessed e.g. after a merging of knowledge bases. Consider, for example, an ontology about marathon runs containing the axiom Healthy $\sqsubseteq$ MarathonParticipant which is supposed to say that somebody (i.e. a person who has signed up for a run) participates in a marathon if he checks out to be healthy. The axiom is reasonable if the domain is for the management of marathon participants' data only. Now imagine that this ontology is merged with other sports knowledge bases, e.g. a boxing domain. It is wrong to infer that every healthy boxer will participate in the marathon, so the original axiom will likely lead to contradictions. We propose to handle this kind of information by modelling the axiom as material inclusion, i.e. as Healthy $\mapsto$ MarathonParticipant, which will indeed not infer participation from a positive health status. However, the weak form of contraposition reasoning featured by material inclusion results in the following situation: If an individual is not known to be contained in MarathonParticipant, then it is known to be not Healthy, resulting in a possible contradiction on health status while avoiding contradiction in terms of marathon participation, which may be preferred in the domain. Material inclusion may thus propagate contradictory information backwards (to the antecedent), while internal inclusion may propagate contradictory information forward (to the consequent).

We remark here that different inclusion axioms provide ontology engineers with a flexible way to define different ontologies according to the intuition explained above. In case only one kind of inclusion shall be used, we recommend to use strong inclusion, as it should serve the ontology engineer's original intention most closely. To give an example, consider the inconsistent subontology of BuggyPolicy[2] (with additional assertions) which says "*GeneralReliabilityUsernamePolicy* (G for short) is a subset of *Reliable*, G and *Messaging* are disjoint, *Reliable* is a subset of *Messaging*, p_1 is an individual of G and p_2 is an individual of $Reliable$". Using strong inclusion results

[2] http://www.mindswap.org/2005/debugging/ontologies/

in the ontology $\{R \rightarrow M, G \rightarrow R, M \rightarrow \neg \mathbf{G}, G \rightarrow \neg M, G(p_1), R(p_2)\}$, where we use obvious abbreviations for the class names. Under the semantics of strong inclusion, $M(p_1), R(p_1), M(p_2), \neg G(p_1)$, and $\neg M(p_1)$ hold, but $G(p_2)$ does not hold. This example shows that our four-valued semantics can give meaningful answers when an ontology is inconsistent, while classical semantics fails to do so.

4 Transforming $\mathcal{ALC}4$ to $\mathcal{ALC}$

It is a pleasing property of $\mathcal{ALC}4$, that it can be translated easily into classical $\mathcal{ALC}$, such that paraconsistent reasoning can be simulated using standard $\mathcal{ALC}$ reasoning algorithms. We briefly present the translation, a preliminary report has appeared in [6].[3]

For any given concept C, its transformation $\overline{C}$ is the concept obtained from C by the following inductively defined transformation.

- If $C = A$ for A an atomic concept, then $\overline{C} = A^+$, where A^+ is a new concept;
- If $C = \neg A$ for A an atomic concept, then $\overline{C} = A^-$, where A^- is a new concept;
- If $C = \top$, then $\overline{C} = \top$;
- If $C = \bot$, then $\overline{C} = \bot$;
- If $C = E \sqcap D$ for concepts D, E, then $\overline{C} = \overline{E} \sqcap \overline{D}$;
- If $C = E \sqcup D$ for concepts D, E, then $\overline{C} = \overline{E} \sqcup \overline{D}$;
- If $C = \exists R.D$ for D a concept and R is a role, then $\overline{C} = \exists R.\overline{D}$;
- If $C = \forall R.D$ for D a concept and R is a role, then $\overline{C} = \forall R.\overline{D}$;
- If $C = \neg\neg D$ for a concept D, then $\overline{C} = \overline{D}$;
- If $C = \neg(E \sqcap D)$ for concepts D, E, then $\overline{C} = \overline{\neg E} \sqcup \overline{\neg D}$;
- If $C = \neg(E \sqcup D)$ for concepts D, E, then $\overline{C} = \overline{\neg E} \sqcap \overline{\neg D}$;
- If $C = \neg(\exists R.D)$ for D a concept and R is a role, then $\overline{C} = \forall R.\overline{\neg D}$;
- If $C = \neg(\forall R.D)$ for D a concept and R is a role, then $\overline{C} = \exists R.\overline{\neg D}$;

Based on this, axioms are transformed as follows, where C_1, C_2 are concepts.

- $\overline{C_1 \mapsto C_2} = \neg\overline{\neg C_1} \sqsubseteq \overline{C}_2$
- $\overline{C_1 \sqsubset C_2} = \overline{C_1} \sqsubseteq \overline{C}_2$;
- $\overline{C_1 \rightarrow C_2} = \{\overline{C_1} \sqsubseteq \overline{C}_2, \overline{\neg C_2} \sqsubseteq \overline{\neg C_1}\}$.
- $\overline{C(a)} = \overline{C}(a), \overline{R}(a, b) = R(a, b)$, where a, b are individuals, C a concept, R a role.

The following theorem shows that paraconsistent reasoning can indeed be simulated on standard reasoners by means of the transformation just given.

Theorem 3 *For any ontology O we have $O \models_4 \alpha$ if and only if $\overline{O} \models_2 \overline{\alpha}$, where $\models_2$ is the entailment in classical $\mathcal{ALC}$.*

We note that the transformation algorithm is linear in the size of the ontology. This implies that paraconsistent reasoning in our paradigm is not more expensive than classical reasoning.

[3] In [6], it was actually spelled out for $\mathcal{SHOIN}$.

5 Resolution-based Reasoning with $\mathcal{ALC}4$

There exist two fundamentally different approaches to reasoning with description logics. The first, historic approach is based on an adaptation of the tableaux method from first-order predicate logic (see [10]), and is implemented in most current reasoners. The second approach is based on resolution and has been realised in the KAON2 reasoner [9]. While the first method invokes a classical reasoner as a black-box by a preprocessing spelled out in Section 4, the paraconsistent resolution given in this section views the classical reasoner KAON2 as a glass-box, thus avoiding the preprocessing step. We basically follow [9, Chapter 4], and indeed we have to assume that the reader is familiar with the KAON2-approach because space restrictions do not allow us to spell everything out in detail.

We first note that resolution relies heavily on the *tertium non datur*, and thus does not lend itself easily to a paraconsistent setting. In particular, resolution cannot be based on the negation present in paraconsistent logics, as in this case $A \vee B$ and $\neg A \vee C$ does not imply $B \vee C$. We thus start by introducing a second kind of negation, called the *total negation*, denoted by $\sim$. In order to avoid confusion, we will refer to the standard negation as *paraconsist negation*.

Definition 2 *The* total negation $\sim$ *on* $\{\langle P, N\rangle \mid P, N \subseteq \Delta\}$ *is defined by*

$$\sim\langle P, N\rangle = \langle \Delta \setminus P, \Delta \setminus N\rangle.$$

The intuition behind total negation is to reverse both the information of being true and of being false. Notice that we do not extend our four-valued DLs to have the total negation as a concept constructor. We rather use it only to provide a resolution-based decision procedure for four-valued DLs.

Proposition 4 *For total negation, the following hold for all concepts C, D and roles R. For any four-valued interpretation I,*

$$(\sim\sim C)^I = C^I, (\sim\top)^I = \bot^I, (\sim\bot)^I = \top^I, (\neg\sim C)^I = (\sim\neg C)^I$$
$$(\sim(C \sqcup D))^I = (\sim C \sqcap \sim D)^I, (\sim(C \sqcap D))^I = (\sim C \sqcup \sim D)^I,$$
$$(\sim(\forall R.C))^I = (\exists R.\sim C)^I, (\sim(\exists R.C))^I = (\forall R.\sim C)^I.$$

A second issue which we have to address when adjusting resolution to the paraconsistent setting, is to obtain a representation of internal implication (i.e. of internal inclusion) in terms of clauses. We have already remarked in Section 2.2 that internal implication cannot be represented by means of the connectives conjunction, disjunction and paraconsistent negation. However, with total negation a representation of $C \sqsubset D$ as $\sim C \sqcup D$ is possible. The representation is actually not logically equivalent, but it is equisatisfiable, which suffices for setting up a resolution procedure. We indeed have the following theorem.

Theorem 5 *Let O be a four-valued $\mathcal{ALC}4$ ontology, C, D be concepts, I be an interpretation and ι be a new individual not occurring in O. Then the following hold.*

1. $(C \sqsubset D)^I \in \{t, \ddot{\top}\}$ *if and only if* $(\sim C \sqcup D)^I \in \{t, \ddot{\top}\}$.
2. $O \models_4 C(a)$ *if and only if* $O \cup \{\sim C(a)\}$ *is four-valued unsatisfiable.*
3. $O \models_4 C \mapsto D$ *if and only if* $O \cup \{\sim(\neg C \sqcup D)(\iota)\}$ *is four-valued unsatisfiable.*
4. $O \models_4 C \sqsubset D$ *if and only if* $O \cup \{(C \sqcap \sim D)(\iota)\}$ *is four-valued unsatisfiable.*
5. $O \models_4 C \rightarrow D$ *if and only if* $O \cup \{(C \sqcap \sim D)(\iota), (\neg D \sqcap \sim\neg C)(\iota)\}$ *is four-valued unsatisfiable.*

5.1 Translating $\mathcal{ALC}4$ into Clauses

Resolution-based calculi operates on sets of clauses in normal form, so we introduce next clausal forms for $\mathcal{ALC}4$ expressions. We were inspired by [13]. We first define a negation normal form for $\mathcal{ALC}4$ concepts, which we call quasi-NNF.

Definition 3 *A concept C is a quasi-atom, if it is an atomic concept, or in form $\neg A$ where A is an atomic concept. A concept C is a quasi-literal, if it is a quasi-atomic concept, or in form $\sim L$ where L is a quasi-atomic concept. A concept C is in quasi-NNF, if the total negation $\sim$ occurs only in front of quasi-literals.*

To give an example, let A, B, and C be atomic concepts. Then $(A \vee \sim\neg B) \sqcup \forall R.(\sim C)$ is in quasi-NNF. By propositions 2 and 4, the following is obvious.

Theorem 6 *All $\mathcal{ALC}4$ concepts can be transformed into equivalent expressions in quasi-NNF.*

We next define the Definitorial form of $\mathcal{ALC}4$ concepts, which is a technicality to control the size of clauses (see to [9] for details). If C is a concept, then we set

$$\text{Def}(C) = \begin{cases} \{C\} & \text{if } C \text{ is a literal concept,} \\ \{\sim Q \sqcup C|_p\} \cup \text{Def}(C[Q]|_p) & \text{if } p \text{ is eligible for replacement in } C. \end{cases}$$

where $C|_p$ is the position p in concept C, as defined in [14, 9].

As an example, we have $\text{Def}(A \sqcap \exists R.(A \sqcap B)) = \{A \sqcap \exists R.Q, \sim Q \sqcup (A \sqcap B)\}$. Note that $\sim Q \sqcup (A \sqcap B)$ can be interpreted as internal inclusion $Q \sqsubset (A \sqcap B)$, which allows us to use Q as a new name for $(A \sqcap B)$ in $\exists R.(A \sqcap B))$.

The following proposition is as expected.

Proposition 7 *For an $\mathcal{ALC}4$ concept C in quasi-NNF, $C(x)$ has information to be true for all individuals x if and only if all concepts $D_i(x)$ with $D_i \in$ Def(C) have the information to be true. Formally, $\{\top \sqsubset C\}$ is four-valued satisfiable iff $\{\top \sqsubset$ Def$(D_i) \mid D_i \in$ Def$(C)\}$ is.*

We next translate the concepts into predicate logic. This is done by the standard translation as e.g. spelled out in [9] in terms of the function π_y – we just have to provide for the total negation. This is done by allowing the total negation to occur in the predicate logic formulae as well, and by translating total negation in the same way as paraconsistent negation. We make one exception, namely for unversal restriction, where we set $\pi_y(\forall R.C, x) = \forall y.(\sim R(x, y) \sqcup C(y))$.

Table 5. Clause Types

1	$\bigvee(\sim)(\neg)C_i(x) \vee R(x, f(x))$
2	$\bigvee(\sim)(\neg)C_i(x) \vee (\sim)(\neg)D(f(x))$
3	$\bigvee(\sim)(\neg)C_i(x)$
4	$\bigvee(\sim)(\neg)C_i(x) \vee R(x, y) \vee (\sim)(\neg)D(y)$
5	$(\sim)(\neg)C(a)$
6	$(\sim)(\neg)R(a, b)$

Following the above transformations step by step, any $\mathcal{ALC}4$ concept can be translated into a set of first order predicate logic clauses (with total negation) in polynomial size of the original concepts.

The obtained predicate logic formulae (with total negation) can now be translated into clauses in the standard way, i.e. by first casting them into Skolem form, and then into conjunctive normal form by exhaustive application of well-known logical equivalences (see e.g. [14]), which are adjusted for total negation in the obvious straightforward way.

If C is a concept (where the additional use of total negation is allowed), then we denote by $\text{Cls}(C)$ the set of clauses which is obtained by the just mentioned transformation of C. These clauses are predicate logic formulae (with total negation).

We finally translate an $\mathcal{ALC}4$ knowledge base KB into a set $\Xi(KB)$ of predicate logic clauses (with total negation), as follows. The knowledge base $\Xi(KB)$ is the smallest set satisfying the following conditions:

- For each $ABox$ axiom α in $ABox$, $\text{Cls}(\alpha) \subseteq \Xi(KB)$
- For each $TBox$ axiom $C \mapsto D$ in $TBox$, $\text{Cls}(\neg C \sqcup D) \subseteq \Xi(KB)$
- For each $TBox$ axiom $C \sqsubset D$ in $TBox$, $\text{Cls}(\sim C \sqcup D) \subseteq \Xi(KB)$
- For each $TBox$ axiom $C \rightarrow D$ in $TBox$, $\text{Cls}(\sim C \sqcup D, \sim\neg D \sqcup \neg C) \subseteq \Xi(KB)$

Theorem 8 *Let KB be an $\mathcal{ALC}4$ ontology. Then, the following hold.*

- *KB is satisfiable iff $\Xi(KB)$ is satisfiable.*
- *$\Xi(KB)$ can be computed in time polynomial in $|KB|$.*
- *Each clause in $\Xi(KB)$ is of one of the syntactic forms listed in Table 5. We refer to these clauses as* 4-valued clauses*.*

5.2 Ordered Resolution with Selection Function $O4_{DL}$ for $\mathcal{ALC}4$

The KAON2 approach for $\mathcal{ALC}$ is based on a modification of the original resolution calculus, known as ordered resolution (see [9] for the necessary preliminaries). We will now define the corresponding notions of clause ordering and of selection function, which we require for this. We assume in the sequel that all 4-valued clauses are of the form described in theorem 8.

Given any fixed ordering $\succ$ on ground quasi-atoms which is total and well-founded, we can obtain an *ordering on sets of clauses* as follows.

1. Extend $\succ$ to an ordering $\succ_L$ on ground literals by setting $\sim A \succ_L A$ for any A, and $[\sim]A \succ_L [\sim]B$, if $A \succ B$.
2. Extend $\succ_L$ to an ordering $\succ_C$ on ground clauses by setting $\succ_C = (\succ_L)_{mul}$ to be the multi-set extension of $\succ_L$ (see [9] for formal definition). The intuition is that $C_1 \succ_C C_2$ iff the maximal quasi-literal in clause C_1 is greater then that in clause C_2 w.r.t. $\succ_L$.

By a slight abuse of notation, we use $\succ$ also for $\succ_L$ and $\succ_C$ where the meaning is clear from the context.

By a *selection function* we mean a mapping S that assigns to each clause C a (possibly empty) multiset $S(C)$ of literals with the prefix $\sim$ in C. For example, both $\{\sim\neg A\}$ and $\{\sim\neg A, \sim D\}$ can be selected in clause $\sim\neg A \vee \sim D \vee B \vee \neg C$.

An ordered resolution step with selection function can now be described by the inference rule

$$\frac{C \vee A \quad D \vee \sim B}{C\sigma \vee D\sigma}$$

where

- $\sigma = MGU(A, B)$ is the most general unifier of the quasi-atoms A, B, and C, D are quasi-clauses.
- $A\sigma$ is strictly maximal in $C\sigma \vee A\sigma$, and no literal is selected in $C\sigma \vee A\sigma$;
- $\sim B\sigma$ is either selected in $D\sigma \vee \sim B\sigma$, or it is maximal in $D\sigma \vee \sim B\sigma$ and no literal is selected in $D\sigma \vee \sim B\sigma$.

The corresponding ordered factorization rule is

$$\frac{C \vee A \vee B}{(C \vee A)\sigma}$$

where $\sigma = MGU(A, B)$ and $A\sigma$ is maximal in $C\sigma \vee A\sigma$ and nothing is selected in C.

Theorem 9 *(Soundness and Completeness of $O4_{DL}$) Let N be an $\mathcal{ALC}4$ knowledge base. Then $\Xi(N) \vdash_{O4_{DL}} \square$ iff N is four-valued unsatisfiable.*

We can now select suitable parameters in order to arrive at a decision procedure based on $O4_{DL}$. This can be done as follows.

- The literal ordering $\succ$ is defined such that $R(x, f(x)) \succ \sim C(x)$ and $D(f(x)) \succ \sim C(x)$, for all function symbols f, and predicates R, C, and D.
- The selection function selects every binary literal which is preceeded by $\sim$.

Theorem 10 *(Decidability) For an $\mathcal{ALC}4$ knowledge base KB, saturating $\Xi(KB)$ by $\mathcal{O}4_{DL}$ decides satisfiability of KB and runs in time exponential in $|KB|$.*

6 Implementation

ParOWL[4] is a prototype implementation of our paraconsistent reasoning approach. It realises the algorithm from Section 4 by means of a command line tool based on KAON2.

[4] http://logic.aifb.uni-karlsruhe.de/wiki/Paraconsistent_reasoning

Table 6. Paraconsistent reasoning examples

Ontology	Bird	FlyAnimal	Penguin	notBird	notFlyAnimal	notPenguin
O_1	$\emptyset$	$\emptyset$	tweety	$\emptyset$	$\emptyset$	$\emptyset$
O_2	tweety	tweety	tweety	$\emptyset$	tweety	$\emptyset$
O_3	tweety	tweety	tweety	tweety	tweety	tweety
O_4	tweety	$\emptyset$	tweety	$\emptyset$	tweety	$\emptyset$

In order to allow the use of standard OWL syntax, the tool expects four input files as parameters, all of which are standard OWL documents. In the first file, class inclusion is interpreted as material inclusion, in the second as internal inclusion, and in the third as strong inclusion. The fourth file is expected to contain an ABox. ParOWL outputs an OWL file which contains the translation.

To display the usage of the different inclusion axioms in ParOWL, consider the following example ontology O, which consists of the axioms Bird $\sqsubseteq$ FlyAnimal, Penguin $\sqsubseteq$ Bird, Penguin $\sqsubseteq \neg$FlyAnimal, and Penguin(tweety), which is obviously inconsistent. We compare the following different four-valued ontologies which can be derived from O: For O_1 all inclusions are material, for O_2 all inclusions are internal, for O_3 all inclusions are strong. For O_4, the first inclusion is material, the second is internal, and the third is strong. Table 6 shows the extensions of the concepts in these ontologies as computed with ParOWL. The desired result may be O_4, and indeed the choice of inclusion axioms in this case follows the intuitions laid out in Section 3.

7 Conclusions and Further Work

We have motivated and formally described an approach for paraconsistent reasoning with ontologies, which is based on the simultaneous use of different kinds of paraconsistent inclusion. We have provided guidelines for the use of these different inclusions. We have provided algorithms for implementing our approach and presented a publicly available tool which realizes it.

Concerning the two algorithms provided in Sections 4 and 5, it is rather apparent that all the benefits from the KAON2 system – like the ability to handle large ABoxes – can also be achieved by invoking KAON2 after employing the transformation algorithm from Section 4 in a preprocessing manner using ParOWL. However, although the transformation algorithm is polynomial, it may be time consuming for large ontologies. This can possibly be improved by the direct algorithm from Section 5. Most of the technical details of the KAON2 implementation can indeed be carried over to our algorithm from Section 5.

In the literature, there are basically two other approaches to four-valued description logics, namely [4] and [5]. Our approach differs from theirs in two important aspects. The first is that we allow for simultaneous usage of different inclusions. The second is that we propose a translation from our logic into standard description logic such that established reasoners can be used. We thus benefit directly from the highly optimized systems currently available. The logics in [4, 5] have neither of these features.

Obviously, much work remains to be done to make our approach fit for practice. Besides the obvious task of providing a better implementation than just a prototype, we also have to address in more detail the question, which kinds of paraconsistent inclusion are to be chosen when translating paraconsistent ontologies to standard ontologies. We envision a combination of system recommendations with user interactions. Alternatively, inclusions could be weakened gradually from strong inclusion to weaker versions – probably involving even further notions of inclusion – until a reasonable answer to a query is found. These issues are currently under investigation by the authors.

References

1. Hayes, P., Horrocks, I., Patel-Schneider, P.F.: OWL Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation 10 February 2004 (2004)
2. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In Gottlob, G., Walsh, T., eds.: IJCAI, Morgan Kaufmann (2003) 355–362
3. Haase, P., van Harmelen, F., Huang, Z., Stuckenschmidt, H., Sure, Y.: A framework for handling inconsistency in changing ontologies. In Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A., eds.: International Semantic Web Conference. Volume 3729 of Lecture Notes in Computer Science., Springer (2005) 353–367
4. Patel-Schneider, P.F.: A four-valued semantics for terminological logics. Artificial Intelligence **38** (1989) 319–351
5. Straccia, U.: A sequent calculus for reasoning in four-valued description logics. In Galmiche, D., ed.: TABLEAUX. Volume 1227 of Lecture Notes in Computer Science., Springer (1997) 343–357
6. Ma, Y., Lin, Z., Lin, Z.: Inferring with inconsistent OWL DL ontology: A multi-valued logic approach. In Grust, T., et al., eds.: EDBT Workshops. Volume 4254 of Lecture Notes in Computer Science., Springer (2006) 535–553
7. Belnap, N.D.: A useful four-valued logic. Modern uses of multiple-valued logics (1977) 7–73
8. Arieli, O., Avron, A.: The value of the four values. Artif. Intell. **102** (1998) 97–141
9. Motik, B.: Reasoning in description logics using resolution and deductive databases. PhD theis, University Karlsruhe, Germany (2006)
10. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
11. Horrocks, I., Patel-Schneider, P.F.: Reducing OWL entailment to description logic satisfiability. J. Web Sem. **1** (2004) 345–357
12. Arieli, O., Avron, A.: Reasoning with logical bilattices. Journal of Logic, Language and Information **5** (1996) 25–63
13. Kamide, N.: Foundations of paraconsistent resolution. Fundamenta Informaticae **71** (2006) 419–441
14. Fitting, M.: First-Order Logic and Automated Theorem Proving, 2nd Edition. Texts in Computer Science. Springer (1996)