

# Paraconsistent Resolution for Four-valued Description Logics <sup>\*</sup>

Yue Ma<sup>1,2</sup>, Pascal Hitzler<sup>2</sup>, and Zuoquan Lin<sup>1</sup>

<sup>1</sup>Department of Information Science, Peking University, China

<sup>2</sup>AIFB, Universität Karlsruhe, Germany

{mayue,lz}@is.pku.edu.cn, {yum,hitzler}@aifb.uni-karlsruhe.de

**Abstract.** In this paper, we propose an approach to translating any  $\mathcal{ALC}$  ontology (possible inconsistent) into a logically consistent set of disjunctive datalog rules. We achieve this in two steps: First we give a simple way to make any  $\mathcal{ALC}$  based ontology 4-valued satisfiable, and then we study a sound and complete paraconsistent ordered-resolution decision procedure for our 4-valued  $\mathcal{ALC}$ . Our approach can be viewed as a paraconsistent version of KAON2 algorithm.

## 1 Introduction

The study of inconsistency handling in description logics can be divided into two fundamentally different approaches. 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 [1]. The other approach yields to the insight that inconsistencies are a natural phenomenon in realistic data which are to be handled by a logic, such as paraconsistent logics which tolerates it [2–6]. Comprised with the former, the latter acknowledges and distinguishes the different epistemic statuses between "the assertion is true" and "the assertion is true with conflict". In this paper, following [6], we study an approach to reasoning with an inconsistent  $\mathcal{ALC}$  ontology, which belongs to the second.

Considering applications of DLs, the reasoning algorithm is as important as the semantics definition. Compared to the algorithm implemented in [6] by employing a polynomial transformation algorithm which still may be time consuming for large ontologies, the process described in this paper which translate an ontology directly into a satisfiable set of rules saves the preprocessing time. Compared to the algorithm proposed in [4] which is a sequence calculus based procedure, our algorithm can benefit directly from the technical details of the KAON2 implementation. Compared to the work given in [3] where a tractable subsumption is discussed, our approach spells out for both subsumption and instant checking reasoning tasks.

There exist two fundamentally reasoning algorithms which have been implemented in DLs reasoners. The first historic approach is based on tableaux algorithms [7]. The

---

<sup>\*</sup> 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/>), by the Deutsche Forschungsgemeinschaft (DFG) in the ReaSem project, and by China Scholarship Council.

second approach is based on basic superposition (ordered resolution for  $\mathcal{ALC}$ ) realized by KAON2 reasoner [8]. Our algorithm is based on an adaptation of the algorithm underlying KAON2 for dealing with  $\mathcal{ALC4}$ , a paraconsistent  $\mathcal{ALC}$ . Theoretically, we design a paraconsistent ordered-resolution for  $\mathcal{ALC4}$  which is different from other algorithms in [3, 4] and which provides a way to extend KAON2 algorithm to reason with inconsistent ontologies. Due to space limitations, proofs are omitted. They can be found in a technical report under <http://www.aifb.uni-karlsruhe.de/WBS/phi/pub/parowltr.pdf>.

The paper is structured as follows. In Section 2, we review briefly the syntax and semantics of the paraconsistent description logic defined in our previous work [6], where more technical details and intuitions can be found. Section 3 gives the paraconsistent resolution decision procedure and Section 4 studies how a set of consistent disjunctive datalog rules can be obtained from a four-valued  $\mathcal{ALC}$  ontology. Finally we conclude this paper in Section 5.

## 2 The Four-valued Description Logic $\mathcal{ALC4}$

### 2.1 Syntax and Semantics

Syntactically,  $\mathcal{ALC4}$  hardly differs from  $\mathcal{ALC}$ . Complex concepts and assertions are defined in exactly the same way. However, we allow three kinds of class inclusions, corresponding to the three implication connectives in four-valued logic case. They are called material inclusion axiom, internal inclusion axiom, and strong inclusion axiom, denoted as  $C \mapsto D$ ,  $C \sqsubset D$ , and  $C \rightarrow D$ , respectively.

Semantically, four-valued interpretations map individuals to elements of the domain of the interpretation, as usual. For concepts, however, to allow for reasoning with inconsistencies, a four-valued interpretation over domain  $\Delta^I$  assigns to each concept  $C$  a pair  $\langle P, N \rangle$  of (not necessarily disjoint) subsets of  $\Delta^I$ . 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  $C$ .  $P$  and  $N$  are not necessarily disjoint and mutual complementary with respect to the domain.

Formally, a four-valued interpretation is a pair  $I = (\Delta^I, \cdot^I)$  with  $\Delta^I$  as domain, where  $\cdot^I$  is a function assigning elements of  $\Delta^I$  to individuals, and subsets of  $(\Delta^I)^2$  to concepts, such that the conditions in Table 1 are satisfied, where functions  $\text{proj}^+(\cdot)$  and  $\text{proj}^-(\cdot)$  are defined by  $\text{proj}^+\langle P, N \rangle = P$  and  $\text{proj}^-\langle P, N \rangle = N$ .

For the semantics defined above, we ensure that a number of useful equivalences from classical DLs, such as double negation law and Demorgan Law, hold, see [6] for details.

The semantics of the three different types of inclusion axioms is formally defined in Table 2 (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.

**Table 1.** 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 $R_P, R_N \subseteq \Delta^I \times \Delta^I$
$o$	$o^I \in \Delta^I$
$\top$	$\langle \Delta^I, \emptyset \rangle$
$\perp$	$\langle \emptyset, \Delta^I \rangle$
$C_1 \sqcap C_2$	$\langle P_1 \cap P_2, N_1 \cup N_2 \rangle$ , 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 $C^I = \langle P, N \rangle$
$\exists R.C$	$\{x \mid \exists y, (x, y) \in \text{proj}^+(R^I) \text{ and } y \in \text{proj}^+(C^I)\},$ $\{x \mid \forall y, (x, y) \in \text{proj}^+(R^I) \text{ implies } y \in \text{proj}^-(C^I)\}$
$\forall R.C$	$\{x \mid \forall y, (x, y) \in \text{proj}^+(R^I) \text{ implies } y \in \text{proj}^+(C^I)\},$ $\{x \mid \exists y, (x, y) \in \text{proj}^+(R^I) \text{ and } y \in \text{proj}^-(C^I)\}$

**Table 2.** 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	$C_1 \sqsubseteq C_2$	$\text{proj}^+(C_1^I) \subseteq \text{proj}^+(C_2^I)$
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	$R(a, b)$	$(a^I, b^I) \in \text{proj}^+(R^I)$

## 2.2 The satisfiability of $\mathcal{ALC}4$ ontologies

Note that the four-valued semantics given in previous subsection does not assure that every  $\mathcal{ALC}4$  ontology has 4-valued models.<sup>1</sup> The following example illustrates this.

**Example 1** Consider  $\mathcal{T} = \{\top \sqsubseteq \perp\}$ . Since for any four-valued interpretation  $I$ ,  $\top^I = \langle \Delta^I, \emptyset \rangle$  and  $\perp^I = \langle \emptyset, \Delta^I \rangle$ , where  $\Delta^I \neq \emptyset$  for DL interpretations,  $\mathcal{T}$  has no four-valued model according to Table 2. The conclusion remains the same if other two kinds of inclusion are used in  $\mathcal{T}$ .

To make every ontology 4-valued satisfiable, we introduce the following substitution (Definition 1). The underlying intuition is that  $\perp \equiv A \sqcap \neg A$  and  $\top \equiv A \sqcup \neg A$  hold for any concept  $A$  under the classical semantics w.r.t. any ontology.

**Definition 1** Given an ontology  $O$ , the satisfiable form of  $O$ , denote  $SF(O)$ , is the ontology obtained by replacing each occurrence of  $\perp$  in  $(O)$  with  $NA \sqcap \neg NA$ , and replacing each occurrence of  $\top$  in  $(O)$  with  $NA \sqcup \neg NA$ , where  $NA$  is a new atomic concept.

<sup>1</sup> The problem also exists for the 4-valued DL defined in [4] if  $\top$  and  $\perp$  are used arbitrarily.

**Example 2** (Example 1 Continued) By Definition 1,  $SF(\mathcal{T}) = \{NA \sqcup \neg NA \sqsubseteq NA \sqcap \neg NA\}$ . Obviously,  $SF(\mathcal{T})$  has a 4-valued model  $I = \langle \Delta^I, \cdot^I \rangle$ , where  $NA^I = \langle \Delta^I, \Delta^I \rangle$ .  $I$  is also a model of  $\{NA \sqcup \neg NA \mapsto NA \sqcap \neg NA\}$  and  $\{NA \sqcup \neg NA \rightarrow NA \sqcap \neg NA\}$ . That is, a 4-valued unsatisfiable ontology becomes 4-valued satisfiable.

Following proposition says the substitution doesn't impact the classical inconsistency.

**Proposition 1** For any ontology  $O$ , the following two claims hold:

- (1)  $SF(O)$  is 2-valued consistent if and only if  $O$  is 2-valued consistent.
- (2)  $SF(O)$  always has at least one 4-valued model.

Note that claim 2 in proposition 1 doesn't hold for 2-valued semantics, so we cannot expect to make an inconsistent ontology 2-valued satisfiable in the same way. Because of this proposition, we assume that all ontologies discussed in the rest have 4-valued models.

### 3 Resolution-based Reasoning with $\mathcal{ALC4}$

In this section, we basically follow [8, Chapter 4] to study a paraconsistent resolution for  $\mathcal{ALC4}$ , and indeed we have to assume that the reader is familiar with the KAON2-approach because of space restrictions.

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 *total negation*, written  $\sim$ . To avoid confusion, we will refer to  $\neg$  as *paraconsistent negation*. 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.

**Definition 2** 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. By Definition 2, the double negation elimination law and Demogen Laws also hold for total negation  $\sim$ .

An important reason to propose the total negation is that it provides a way to reduce 4-valued entailment relation  $\models_4$  to 4-valued satisfiability, because for any ontology  $O$  and an axiom  $\alpha$ ,  $O \models_4 \alpha$  if and only if  $O \cup \{\sim\alpha\}$  is 4-unsatisfiable. Moreover, total negation can be used to obtain a representation of internal inclusion in terms of clauses, where equisatisfiability retains, because  $(C \sqsubseteq D)^I \in \{t, \ddot{\top}\}$  if and only if  $(\forall x. (\sim C \sqcup D)(x))^I \in \{t, \ddot{\top}\}$ . By these two points, we have following theorem shows that 4-entailment can be converted into 4-unsatisfiability.

**Theorem 2** Let  $O$  be a four-valued  $\mathcal{ALC4}$  ontology,  $C, D$  be concepts,  $I$  be an interpretation and  $\iota$  be a new individual not occurring in  $O$ . Then the following hold.

1.  $O \models_4 C(a)$  if and only if  $O \cup \{\sim C(a)\}$  is four-valued unsatisfiable.
2.  $O \models_4 C \mapsto D$ ,  $O \models_4 C \sqsubset D$ ,  $O \models_4 C \rightarrow D$  if and only if  $O \cup \{\sim(\neg C \sqcup D)(\iota)\}$ ,  $O \cup \{(C \sqcap \sim D)(\iota)\}$ , and  $O \cup \{(C \sqcap \sim D)(\iota), (\neg D \sqcap \sim \neg C)(\iota)\}$  are four-valued unsatisfiable, respectively.

### 3.1 Translating $\mathcal{ALC}4$ into Clauses

To introduce clausal forms for  $\mathcal{ALC}4$  expressions, we first define an extended negation normal form for  $\mathcal{ALC}4$  called quasi-NNF. We are inspired by [9].

**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.  $C$  is a quasi-literal, if it is a quasi-atomic concept, or in form  $\sim L$  where  $L$  is a quasi-atomic concept.  $C$  is in quasi-NNF, if  $\sim$  occurs only in front of quasi-literals and  $\neg$  does not occur in front of  $\sim$ .

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. Based on the properties of  $\sim$ , it is easy to check that all  $\mathcal{ALC}4$  concepts can be transformed into equivalent expressions in quasi-NNF.

We next translate the concepts into predicate logic. This is done by the standard translation as e.g. spelled out in [8] in terms of the function  $\pi_y$  – we just have to provide for the total negation. We make one exception, namely for universal restriction, where we set  $\pi_y(\forall R.C, x) = \forall y.(\sim R(x, y) \sqcup C(y))$ . 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 [10], which are adjusted for total negation in the straightforward way. To avoid the exponential blowup and to preserve the structure of formulae, we also can apply the structural transformation [11] to  $\mathcal{ALC}4$  as used in [8]. To do this, we define the paraconsistent Definitorial Form  $\text{PDef}(\cdot)$  of  $\mathcal{ALC}4$  concepts as follows. Note that total negation should be used.

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

where  $C|_p$  to be the position  $p$  in concept  $C$ , as defined in [10, 8].

**Proposition 3** For an  $\mathcal{ALC}4$  concept  $C$  in quasi-NNF,  $\{\top \sqsubset C\}$  is four-valued satisfiable iff  $\{\top \sqsubset \text{PDef}(D_i) \mid D_i \in \text{PDef}(C)\}$  is.

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. 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 give an example.

**Example 3** The concept  $\neg(\sim A \sqcap \exists R.(\forall S.C))$  is translated as follows.

$$\begin{aligned} & \neg(\sim A \sqcap \exists R.(\forall S.C)) \\ \text{in quasi-NNF:} & \quad \sim \neg A \sqcup \forall R.(\exists S. \neg C) \\ \text{in PDef:} & \quad \{\sim \neg A \sqcup \forall R.Q, \sim Q \sqcup \exists S. \neg C\} \\ \text{in predicate logic:} & \quad \{\sim \neg A(x) \vee \sim R(x, y) \vee Q(y), \sim Q(x) \vee S(x, f(x)), \\ & \quad \quad \quad \sim Q(x) \vee \neg C(f(x))\} \end{aligned}$$

Based on the transformation described above, we finally can translate an  $\mathcal{ALC}4$  ontology  $O$  into a set of predicate logic clauses (with total negation)  $\Xi(O)$  which is the smallest set satisfying the following conditions:

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

**Theorem 4** *Let  $O$  be an  $\mathcal{ALC}4$  ontology.  $O$  is 4-valued satisfiable iff  $\Xi(O)$  is 4-valued satisfiable.*

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

Given any fixed ordering  $\succ$  on ground quasi-atoms which is total and well-founded, we can obtain an *ordering on sets of clauses* in standard way as stated in [8]. By a slight abuse of notation, we use  $\succ$  also for  $\succ_L$  and  $\succ_C$  where the meaning is clear from the context. For example, if  $\neg A \succ A \succ B \succ \neg B \succ D$ , then  $[\sim]\neg A \succ [\sim]A \succ [\sim]B \succ [\sim]\neg[\sim]B \succ [\sim]D$  and  $\sim A \vee \neg B \prec \neg A \vee B \prec \sim \neg A \vee D$ .

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 and Factorization Rule as follows, respectively:

$$\frac{C \vee A \quad D \vee \sim B}{C\sigma \vee D\sigma} \quad \text{and} \quad \frac{C \vee A \vee B}{(C \vee A)\sigma},$$

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  $D\sigma \vee \sim B\sigma$ .

**Theorem 5** *(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.*

Although the inference rules are different from those of  $\mathcal{ALC}$ , we find that similar way of the selection of the literal ordering and selection function still provide us a decision procedure for  $\mathcal{ALC}4$ .

- 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 preceded by  $\sim$ .

**Theorem 6** *(Decidability) For an  $\mathcal{ALC}4$  knowledge base  $KB$ , saturating  $\Xi(KB)$  by  $O4_{DL}$  decides satisfiability of  $KB$  and runs in time exponential in  $|KB|$ .*

## 4 Translating $\mathcal{ALC4}$ to satisfiable Disjunctive Datalog

We should distinguish the total negation and paraconsistent negation during the translation, while there is only one kind of negation in [8]. As the translation from  $\mathcal{ALC}$  to disjunctive datalog, we can perform all inferences among nonground clauses first, after which we can simply delete all nonground clauses containing function symbols. The remaining clause set consists of clauses without function symbols [8].

**Definition 4** For an extensionally reduced  $\mathcal{ALC}$  ontology  $O$ , the function-free version of ontology  $O$  is defined as follows:

$$FF(O) = \lambda(\Gamma(O_T)) \cup \Gamma(O_A) \cup \{HU(a) \mid \text{for each individual } a \text{ occurring in } O\}$$

where  $\Gamma(O_T)$  is the set of clauses obtained by saturating  $\Xi(O_T)$  by  $O_{4DL}$  and then deleting all the clauses containing functions, and  $\Gamma(O_A)$  is the set of clauses obtained by saturating  $\Xi(O_A)$  by  $O_{4DL}$ . And for a clause  $C$ ,  $\lambda(C) = C \cup \{\sim HU(x) \mid \text{for each unsafe variable } x \text{ in } C\}$  and  $\lambda(\Sigma) = \{\lambda(C) \mid \text{for each clause } C \text{ in } \Sigma\}$ .

Note that the total negation is used in the  $\lambda$  operator. Next, we introduce following definition to simplify the statement of the translation process.

**Definition 5** All the literals in  $FF(O)$  are in one of the following cases:

- Pure positive literal which is just an atom (i.e., without neither paraconsistent negation nor total negation in front);
- Paraconsistent negative literal which is constructed by a pure positive literal with the paraconsistent negation occurring in its front.
- Total negative literal which is constructed by a pure positive literal or a paraconsistent negative literal with the total negation occurring in its front.

By this definition and Definition 3, positive quasi-literals include both pure positive literals and paraconsistent negative literals, while total negative literals are very negative quasi-literals. To give an example, let  $A(x)$  be an atom, then it is a pure positive literal,  $\neg A(x)$  is a paraconsistent negative literal, and  $\sim\neg A(x)$ ,  $\sim A(x)$  are total negative literals.

**Definition 6** The disjunctive datalog  $DD_4(O)$  corresponding to an ontology  $O$  is defined by moving in each clause from  $FF(O)$

- each pure positive liter into the rule head;
- each paraconsistent negative literal into the rule head as well and replacing it with a fresh atom simultaneously;
- each total negative literal into the rule body and replacing its quasi-atom part with a fresh atom simultaneously.

If  $KB$  is not extensionally reduced, then  $DD(KB) = DD(KB')$ , where  $KB'$  is an extensionally reduced knowledge base obtained from  $KB$  in the standard way [8].

From section 2.2, we can see that every ontology  $O$  can become 4-valued satisfiable after appropriate rewriting. That is, by 4-valued semantics, meaningful conclusions always can be derived from even a classically inconsistent ontology. The first claim of the following theorem shows that  $DD_4(O)$  is always consistent as well such that untrivial answers can be returned from it.

**Theorem 7** *Let  $KB$  be an  $ALC4$  ontology. Then, the following claims hold:*

1.  $DD_4(\tilde{O})$  is always satisfiable for any ontology  $O$ , where  $\tilde{O}$  is the ontology defined in Definition 1.
2.  $O \models_4 \alpha$  if and only if  $DD_4(O) \models_c \alpha$ , where  $\alpha$  is of the form  $A(a)$  or  $R(a, b)$ , and  $A$  is an atomic concept.

## 5 Conclusions

In this paper, we work out for how to translate an  $ALC4$  ontology to a consistent set of disjunctive datalog rules, such that meaningful consequences can be deduced from a possible inconsistent ontology.

Considering the algorithm described in this paper, it is rather apparent that all the benefits from the KAON2 system – like the ability to handle large ABoxes – can also be achieved by the paraconsistent version of KAON2. We will further study the similar paraconsistent approach for more complex DLs in the future work.

## References

1. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: IJCAI. (2003) 355–362
2. Huang, Z., van Harmelen, F., ten Teije, A.: Reasoning with inconsistent ontologies. In: IJCAI. (2005) 454–459
3. Patel-Schneider, P.F.: A four-valued semantics for terminological logics. Artificial Intelligence **38** (1989) 319–351
4. Straccia, U.: A sequent calculus for reasoning in four-valued description logics. In: TABLEAUX. (1997) 343–357
5. Straccia, U.: Foundations of a logic based approach to multimedia document retrieval. Department of Computer Science, University of Dortmund, Dortmund, Germany (June, 1999)
6. Ma, Y., Hitzler, P., Lin, Z.: Algorithms for paraconsistent reasoning with OWL. In: Proc. of ESWC. (2007) To appear.
7. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
8. Motik, B.: Reasoning in description logics using resolution and deductive databases. PhD thesis, University Karlsruhe, Germany (2006)
9. Kamide, N.: Foundations of paraconsistent resolution. Fundamenta Informaticae **71** (2006) 419–441
10. Fitting, M.: First-Order Logic and Automated Theorem Proving, 2nd Edition. Texts in Computer Science. Springer (1996)
11. Plaisted, D.A., Greenbaum, S.: A structure-preserving clause form translation. J. Symb. Comput. **2** (1986) 293–304