

## Measuring Inconsistency in *DL-Lite* Ontologies

Liping Zhou\*, Houkuan Huang\*, Guilin Qi<sup>†</sup>, Yue Ma<sup>‡</sup>, Zhisheng Huang<sup>§</sup> and Youli Qu\*

\*Beijing Jiaotong University, Beijing, China

Email: dearliping@gmail.com, {hkhuang, ylqu}@bjtu.edu.cn

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

Email: gqi@aifb.uni-karlsruhe.de

<sup>‡</sup>LIPN, Université Paris-Nord, France

Email: Yue.Ma@lipn.univ-paris13.fr

<sup>§</sup>Department of Computer Science, Vrije Universiteit Amsterdam, The Netherlands

Email: huang@cs.vu.nl

**Abstract**—Measuring Inconsistency in ontologies is an important topic in ontology engineering as it can provide extra information for dealing with inconsistency. Many approaches have been proposed to deal with this issue. However, the main drawback of these algorithms is their high computational complexity. One of the main sources of the high complexity is the intractability of the underlying Description Logics (DLs). In this paper, we focus on an important tractable DL family, *DL-Lite*. We define an inconsistency degree of a *DL-Lite* ontology based on a three-valued semantics. We also present an algorithm to compute this inconsistency degree and show that its time-complexity is PTime in the size of ABox and TBox.

**Keywords**—Measuring; Inconsistency Degree; *DL-Lite*; ontologies;

### I. INTRODUCTION

Inconsistencies frequently occur within the ontology life-cycle, such as ontology construction, ontology evolution and ontology merging. Handling inconsistencies, especially, handling logical inconsistency in ontologies is increasingly recognized as an important research topic. When dealing with logical inconsistency (or inconsistency for short), we frequently need extra information that can facilitate us to choose a proper strategy to resolve this problem. It has been shown that measuring inconsistency in ontologies can provide valuable information for many different inconsistency handling approaches, such as revising ontologies [1], [2], debugging ontologies [3], [4] and evaluating inconsistent ontologies [5].

There are some applications for inconsistency measurements.

- Inconsistency measurements enable us to say how “un-valued” an ontology is by showing how inconsistent it is [6]. For example, given two ontologies  $\mathcal{K}_1$  and  $\mathcal{K}_2$ , suppose that the inconsistency degree of  $\mathcal{K}_1$  is less than that of  $\mathcal{K}_2$ , then we can consider that  $\mathcal{K}_1$  is more reliable than  $\mathcal{K}_2$ .
- Inconsistency measurements can also give guidance to resolve inconsistency. When resolving inconsistency, there often have several alternative solutions. It would

be helpful to have some extra information (such as an ordering on axioms of the ontology) to decide which one is the best. For example, we can first rank the axioms in an inconsistent ontology by applying the method in [7], then remove or weaken those axioms with lower priority to restore consistency. If the inconsistency degree is low and repairing it is time-consuming or error prone, we can tolerate it and apply paraconsistent semantics (such as the one given in [8]) to reason with it.

A number of proposals have been made for measuring the inconsistency of ontologies [4], [5], [7], [8]. They can be roughly divided into the following two categories. The first one is to count *the minimal number of formulae* which are responsible for an inconsistency [4]. The second one is to compute *the proportion of language* that is affected by the inconsistencies of ontologies [5], [9]. Our approach belongs to the second category.

Deng et al. [7] provided a method for measuring inconsistency of axioms to identify which axioms need to be removed or modified to resolve an inconsistency. However, their algorithm needs exponential time in the worst case [7]. Qi and Hunter [4] provided a method for measuring *incoherence* of an ontology based on the computation of all the *minimal incoherence-preserving sub-TBox (MIPS)* which is a hard task. For example, it has been shown in [10] that computing all the MIPS of an ontology is NP-hard for tractable DL  $\mathcal{EL}^+$ . Ma et al. [5] proposed a method for measuring inconsistency of a DL  $\mathcal{ALC}$  knowledge base on 4-valued semantics, which can be realized by invoking a DL reasoner [11]. The above approaches to measuring inconsistency are usually based on expressive DLs which suffer from worst-case exponential time behavior of reasoning [1]. This may hinder their applications to ontologies with over large amounts of data.

Recently, there have been some discussions on inconsistency handling in *DL-Lite* (see [12]), an important tractable DL family, which can keep all the reasoning tasks tractable, in particular, with polynomial time complexity with respect

to the size of the ontology [13]. Like other DLs, inconsistencies in *DL-Lite* can also easily occur because disjoint axioms are allowed. The purpose of our research is to investigate *DL-Lite* family to see how to compute the inconsistency degree of a *DL-Lite* ontology in a tractable way.

In this paper, we propose an approach to measuring the inconsistency of a *DL-Lite* ontology based on three-valued semantics. Unlike the approach using the sequence of values to measure inconsistency in [5], we use a single value to measure inconsistency of a *DL-Lite* ontology. To compute the inconsistency degree of a *DL-Lite* ontology based on multi-valued semantics, one way is to list all models w.r.t a specific domain to check preferred models and compute the number of conflicting assertions in such a model. However, listing all models is not an easy reasoning task even for tractable DLs like *DL-Lite*. To alleviate the problem of intractability, we propose a polynomial-time algorithm to compute the inconsistency degree of a *DL-Lite* ontology based on a three-valued semantics by exploring the specific feature of *DL-Lite*. The main contributions of our paper can be summarized as follows:

- We define a three-valued semantics for two important DLs in *DL-Lite* family: *DL-Lite<sub>F</sub>* and *DL-Lite<sub>R</sub>*.
- Given a *DL-Lite* ontology  $O = \langle \mathcal{T}, \mathcal{A} \rangle$ , we show that it is desirable to consider domain  $\Delta^{db(\mathcal{A})}$  and  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$  to measure inconsistency. Then we give a definition of the inconsistency degree of a *DL-Lite* ontology.
- An algorithm is presented to compute a preferred model for  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$  under the three-valued semantics. We show the correctness of our algorithm and demonstrate that its time-complexity is PTime in the size of an ontology.

The rest of the paper is organized as follows. Section II presents some basic notions for *DL-Lite*. Section III gives a three-valued semantics for *DL-Lite*. Section IV introduces our approach to measuring inconsistency for a *DL-Lite* ontology. Section V gives an algorithm to compute the inconsistency degree and analyzes its computational complexity. We conclude our paper in Section VI.

## II. PRELIMINARIES

*DL-Lite* is a family of DLs that aims to capture some of the most popular conceptual modeling formalisms, such as Entity-Relationship model and UML class diagrams, while preserving the tractability of the most important reasoning tasks, such as ontology satisfiability. We mainly consider two important DLs in *DL-Lite* family: *DL-Lite<sub>F</sub>* and *DL-Lite<sub>R</sub>* [13].

The language of *DL-Lite<sub>core</sub>* is the core language for *DL-Lite<sub>F</sub>* and *DL-Lite<sub>R</sub>*, in which concepts and roles are formed according to the following syntax:

$$\begin{array}{l} B \longrightarrow A \mid \exists R \quad R \longrightarrow P \mid P^- \\ C \longrightarrow B \mid \neg B \quad E \longrightarrow R \mid \neg R \end{array}$$

Table I  
SYNTAX AND SEMANTICS OF *DL-Lite*

Syntax	Semantics
$A$	$A^I \subseteq \Delta^I$
$\exists R$	$\{d \mid \exists e, (d, e) \in R^I\}$
$\neg A$	$\Delta^I \setminus A^I$
$\neg \exists R$	$\Delta^I \setminus (\exists R)^I$
$P$	$P^I \in \Delta^I \times \Delta^I$
$P^-$	$\{(o, o') \mid (o', o) \in P^I\}$
$\neg R$	$(\Delta^I \times \Delta^I) \setminus R^I$
$B_1 \sqsubseteq B_2$	$B_1^I \subseteq B_2^I$
$R_1 \sqsubseteq R_2$	$R_1^I \subseteq R_2^I$
(funct $R$ )	$\forall d, e, e', (d, e) \in R^I \wedge (d, e') \in R^I \rightarrow e = e'$
$A(a)$	$a^I \in A^I$
$P(a, b)$	$(a^I, b^I) \in P^I$

where  $A$  and  $P$  denote an atomic concept and an atomic role respectively;  $B$  denotes a *basic concept* (i.e., a concept of the form  $A, \exists R$ );  $R$  denotes a *basic role* (i.e., a role of the form  $P, P^-$ ), where  $P^-$  denotes the inverse of the atomic role;  $C$  denotes a *general concept* (i.e., a concept of the form  $B, \neg B$ ), whereas  $E$  denotes a *general role* (i.e., a concept of the form  $R, \neg R$ ).

A *DL-Lite<sub>core</sub>* TBox is a set of inclusion axioms of the form  $B \sqsubseteq C$ . A *DL-Lite<sub>core</sub>* ABox is a set of membership assertions on atomic concepts and atomic roles:  $A(a), P(a, b)$ , where  $a$  and  $b$  are constants.

*DL-Lite<sub>R</sub>* extends *DL-Lite<sub>core</sub>* with the ability of specifying inclusion assertions between roles of the form  $R \sqsubseteq E$ , where  $R$  and  $E$  are defined as above. *DL-Lite<sub>F</sub>* extends *DL-Lite<sub>core</sub>* with the ability of specifying functionality on roles or on their inverses. Assertions used for this purpose are of the form (funct  $R$ ) and called functionality assertions. Hereinafter, we use the term *DL-Lite* to refer to either *DL-Lite<sub>R</sub>* or *DL-Lite<sub>F</sub>*, we call assertions of the form  $B_1 \sqsubseteq B_2$  or of the form  $R_1 \sqsubseteq R_2$  positive inclusions (PIs), and we call assertions of the form  $B_1 \sqsubseteq \neg B_2$  or  $R_1 \sqsubseteq \neg R_2$  negative inclusions (NIs). The semantics of *DL-Lite* is given by means of an interpretation  $I = (\Delta^I, \cdot^I)$ , consisting of a non-empty *interpretation domain*  $\Delta^I$  and an *interpretation function*  $\cdot^I$  satisfying the conditions in Table I. The function  $\cdot^I$  assigns to each concept  $C$  a subset  $C^I$  of  $\Delta^I$ , and to each role  $R$  a binary relation  $R^I$  over  $\Delta^I$ . An interpretation satisfies a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  (i.e., a model of the ontology) if and only if it satisfies each axiom in both ABox and TBox. An ontology is satisfiable if it has at least one model. An ontology  $\mathcal{K}$  logically implies an assertion  $\alpha$ , written  $\mathcal{K} \models \alpha$ , if all models of  $\mathcal{K}$  are also models of  $\alpha$ . The unique name assumption on constants [14] is adapted by *DL-Lite*. Furthermore, *DL-Lite<sub>R</sub>* has the finite model property, that is, if a *DL-Lite<sub>R</sub>* is consistent, then it has a classical model whose domain is finite [13], [14]. However, *DL-Lite<sub>F</sub>* does not have finite model property [13].

Calvanese et al. [13] have given a novel property about a *DL-Lite* ontology, that is, a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

is satisfiable iff  $db(\mathcal{A})$  is a model of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$ , where  $db(\mathcal{A}) = \langle \Delta^{db(\mathcal{A})}, db(\mathcal{A}) \rangle$  is an interpretation about  $\mathcal{A}$  defined as follows:

- $\Delta^{db(\mathcal{A})}$  is the nonempty set consisting of all constants occurring in  $\mathcal{A}$ ;
- $a^{db(\mathcal{A})} = a$ , for each constant  $a$ ;
- $A^{db(\mathcal{A})} = \{a \mid A(a) \in \mathcal{A}\}$ , for each atomic concept  $A$ ;
- $P^{db(\mathcal{A})} = \{(a_1, a_2) \mid P(a_1, a_2) \in \mathcal{A}\}$ , for each atomic role  $P$ .

$cln(\mathcal{T})$  is the NI-closure of  $\mathcal{T}$  defined as follows:

- All negative inclusions in  $\mathcal{T}$  are also in  $cln(\mathcal{T})$ .
- All functionality assertions in  $\mathcal{T}$  are also in  $cln(\mathcal{T})$ .
- If  $B_1 \sqsubseteq B_2$  is in  $\mathcal{T}$  and  $B_2 \sqsubseteq \neg B_3$  or  $B_3 \sqsubseteq \neg B_2$  is in  $cln(\mathcal{T})$ , then  $B_1 \sqsubseteq \neg B_3$  is in  $cln(\mathcal{T})$ .
- If  $R_1 \sqsubseteq R_2$  is in  $\mathcal{T}$  and  $\exists R_2 \sqsubseteq \neg B$  or  $B \sqsubseteq \neg \exists R_2$  is in  $cln(\mathcal{T})$ , then  $\exists R_1 \sqsubseteq \neg B$  is in  $cln(\mathcal{T})$ .
- If  $R_1 \sqsubseteq R_2$  is in  $\mathcal{T}$  and  $\exists R_2^- \sqsubseteq \neg B$  or  $B \sqsubseteq \neg \exists R_2^-$  is in  $cln(\mathcal{T})$ , then  $\exists R_1^- \sqsubseteq \neg B$  is in  $cln(\mathcal{T})$ .
- If  $R_1 \sqsubseteq R_2$  is in  $\mathcal{T}$  and  $R_2 \sqsubseteq \neg R_3$  or  $R_3 \sqsubseteq \neg R_2$  is in  $cln(\mathcal{T})$ , then  $R_1 \sqsubseteq \neg R_3$  is in  $cln(\mathcal{T})$ .
- In the case in which  $\mathcal{T}$  is a  $DL\text{-lite}_{\mathcal{F}}$  TBox, if one of the assertions  $\exists R \sqsubseteq \neg \exists R$  or  $\exists R^- \sqsubseteq \neg \exists R^-$  is in  $cln(\mathcal{T})$ , then both such assertions are in  $cln(\mathcal{T})$ .
- In the case in which  $\mathcal{T}$  is a  $DL\text{-lite}_{\mathcal{R}}$  TBox, if one of the assertions  $\exists R \sqsubseteq \neg \exists R$ ,  $\exists R^- \sqsubseteq \neg \exists R^-$  or  $R \sqsubseteq \neg R$  is in  $cln(\mathcal{T})$ , then the three such assertions are all in  $cln(\mathcal{T})$ .

In fact,  $cln(\mathcal{T})$  is a special TBox that does not contain PIs and is obtained by closing the NIs with respect to the PIs in  $\mathcal{T}$ .

### III. THREE-VALUED SEMANTICS FOR $DL\text{-Lite}$

To measure the inconsistency of an ontology, we define a three-valued semantics that allows a third truth value *contradictory*. In this way, three-valued semantics provides an approach to define the inconsistency degree of an ontology. Since we only aim to analyze inconsistency, there is no need to adopt other multi-valued semantics, such as four-valued semantics which contains a fourth truth value for expressing incomplete knowledge. In contrast, three-valued semantics is easier to be used because it does not consider the fourth truth value *unknown*.

For a given domain  $\Delta$  and a concept  $C$  (resp., a role  $R$ ), a three-valued interpretation over  $\Delta$  assigns to  $C$  (resp.,  $R$ ) an extended truth value  $\langle C_P, C_N \rangle$  (resp.,  $\langle R_P, R_N \rangle$ ), where  $C_P$  is the subset of  $\Delta$  (resp.,  $R_P$  is the subset of  $\Delta \times \Delta$ ) that supports  $C$  (resp.,  $R$ ) to be true and  $C_N$  is the subset of  $\Delta$  (resp.,  $R_N$  is the subset of  $\Delta \times \Delta$ ) that supports  $C$  (resp.,  $R$ ) to be false [8], and the requirement  $C_P \cup C_N = \Delta$  (resp.,  $R_P \cup R_N = \Delta \times \Delta$ ) must hold under three-valued semantics. We denote  $proj^+(\langle P, N \rangle) = P$  and  $proj^-(\langle P, N \rangle) = N$  [8]. The three-valued semantics of  $DL\text{-Lite}$  is given by means of an interpretation  $I = (\Delta^I, \cdot^I)$

Table II  
THREE-VALUED SEMANTICS OF  $DL\text{-Lite}$

Constructor	Semantics
$C$	$C^I = \langle C_P, C_N \rangle$ , where $C_P, C_N \subseteq \Delta^I$ and $C_P \cup C_N = \Delta^I$
$R$	$R^I = \langle R_P, R_N \rangle$ , where $R_P, R_N \subseteq \Delta^I \times \Delta^I$ and $R_P \cup R_N = \Delta^I \times \Delta^I$
$R^-$	$(R^-)^I = \langle R_P^-, R_N^- \rangle$ , where $R_P^-, R_N^-$ represent the inverse relations on $R_P$ and $R_N$ , respectively.
$\neg C$	$(\neg C)^I = \langle C_N, C_P \rangle$
$\neg R$	$(\neg R)^I = \langle R_N, R_P \rangle$
$\exists R$	$(\exists R)^I = \langle \{x \mid \exists y \in \Delta^I, (x, y) \in R_P^I\}, \{x \mid \forall y \in \Delta^I, (x, y) \in R_N^I\} \rangle$
$\neg \exists R$	$(\neg \exists R)^I = \langle \{x \mid \forall y \in \Delta^I, (x, y) \in R_N^I\}, \{x \mid \exists y \in \Delta^I, (x, y) \in R_P^I\} \rangle$
$=$	$(=)^I = \langle =_P, =_N \rangle$ , where $=_P, =_N \in \Delta^I \times \Delta^I$

consisting of a non-empty *interpretation domain*  $\Delta^I$  and an *interpretation function*  $\cdot^I$  satisfying the conditions in Table II. In Table II, we introduce the three-valued semantics to “=” to represent the three-valued semantics of a functionality assertion. For any given domain  $\Delta$ , we assign to “=” an extended truth value  $\langle =_P, =_N \rangle$ , where “ $=_P$ ” stands for the set of pairs of constants which are equal and “ $=_N$ ” stands for the set of pairs of constants which are not equal. The UNA can be expressed as  $\forall x, y \in \Delta^{db(\mathcal{A})}, (x, y) \in proj^-( (= )^I )$ .

Based on the three-valued semantics, there are three truth values for membership assertions. The three truth values are *true*, *false* and *contradictory*, and we use the symbols  $t, f, \mathbb{B}$  to denote them respectively [5]. The corresponding three-valued semantics for concept assertions is given as follows:

**Definition 1:** [8] For any given instance  $a \in \Delta^I$  and concept name  $A$ ,

- $A^I(a) = t$ , iff  $a \in proj^+(A^I)$  and  $a \notin proj^-(A^I)$ ;
- $A^I(a) = f$ , iff  $a \notin proj^+(A^I)$  and  $a \in proj^-(A^I)$ ;
- $A^I(a) = \mathbb{B}$ , iff  $a \in proj^+(A^I)$  and  $a \in proj^-(A^I)$ .

The corresponding three-valued semantics for role assertion (or equality “ $=$ ”) can be defined in a similar way.

In Table III, we give the three-valued semantics for axioms in  $DL\text{-Lite}$  [15], where  $\Delta \setminus S$  denotes the complementary set of a set  $S$  w.r.t a domain  $\Delta$ . A three-valued model of a  $DL\text{-Lite}$  ontology  $\mathcal{K}$  is a three-valued interpretation  $I$  which satisfies each assertion and each axiom in  $\mathcal{K}$ . A  $DL\text{-Lite}$  ontology is three-valued satisfiable (unsatisfiable) if there exists (does not exist) such a model.

**Example 1:** Given a  $DL\text{-Lite}$  ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , where  $\mathcal{T} = \{PhDStud \sqsubseteq Stud, PhDStud \sqsubseteq Employee, Stud \sqsubseteq \neg Employee, Stud \sqsubseteq \exists hasTutor, (funct\ hasTutor)\}$ ,  $\mathcal{A} = \{PhDStud(a), hasTutor(a, b), hasTutor(a, c)\}$ . We can find that it is an inconsistent ontology. Consider the following three-valued interpretation  $I = (\Delta^I, \cdot^I)$ , where  $\Delta^I = \{a, b, c\}$ ,  $PhDStud^I = \{a\}$ ,  $Stud^I = \{a\}$ ,  $Employee^I = \{a\}$ ,  $hasTutor^I = \{(a, c), (a, b)\}$ ,  $\{(a, a), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c)\}$

Table III  
THREE-VALUED SEMANTICS FOR AXIOMS IN *DL-Lite*

Syntax	Semantics
$B \sqsubseteq C$	$proj^+(B^I) \subseteq proj^+(C^I)$
$R_1 \sqsubseteq R_2$	$proj^+(R_1^I) \subseteq proj^+(R_2^I)$
(functR)	$\forall x, y, z, (x, y) \in proj^+(R^I) \wedge (x, z) \in proj^+(R^I) \rightarrow (y, z) \in proj^+(R^I)$
$A(a)$	$a^I \in proj^+(A^I)$
$P(a, b)$	$(a^I, b^I) \in proj^+(P^I)$
$a = b$	$(a^I, b^I) \in proj^+( (= )^I )$
$a \neq b$	$(a^I, b^I) \in proj^-( (= )^I )$

$\}, (=)^I = \{(a, a), (b, b), (c, c), (b, c)\}, \{(a, b), (a, c), (b, a), (b, c), (c, a), (c, b)\}$ . We can find that  $I$  is a three-valued model of  $\mathcal{K}$  and  $PhDStud^I(a) = t, Stud^I(a) = t, Employee^I(a) = \mathbb{B}$  and  $(=)^I(b, c) = \mathbb{B}$ . It is easy to obtain three-valued semantics for other atomic assertions.

Because of the unique name assumption of the *DL-Lite*, a three-valued interpretation can be a model only if the cardinality of its domain is equal to or greater than the number of constants in an ontology [5]. For a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ ,  $\Delta^{db(\mathcal{A})}$  contains all the constants in  $\mathcal{K}$ . In the following, we only consider those domains whose cardinalities are equal to or greater than that of  $\Delta^{db(\mathcal{A})}$ .

**Proposition 1:** Any *DL-Lite* ontology has the finite model property under three-valued semantics.

Proposition 1 tells us that any *DL-Lite* ontology has at least a three-valued model whose domain is finite. We will use this property to compute preferred models of *DL-Lite* ontologies later.

**Example 2:** Consider a *DL-Lite<sub>F</sub>* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  with  $\mathcal{T} = \{A \sqsubseteq \exists P, \exists P^- \sqsubseteq A, (\text{funct } P^-), B \sqsubseteq \exists P, B \sqsubseteq \neg A\}$  and  $\mathcal{A} = \{B(a)\}$ . It is easy to see that  $\mathcal{K}$  admits only infinite models under classical semantics. However we can give a three-valued model of  $\mathcal{K}$  as follows:  $I = (\Delta^I, \cdot^I)$ , where  $\Delta^I = \{a\}, A^I(a) = \mathbb{B}, B^I(a) = t, P^I(a, a) = t$ .

#### IV. MEASURING INCONSISTENCY

In this section, we will give a formal definition of the inconsistency degree of a *DL-Lite* ontology. We first give some definitions and theorems which will be used to motivate our definition of inconsistency degree.

**Definition 2:** Let  $I$  be a three-valued model of a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  with domain  $\Delta^I$ , and let  $\mathcal{L}_{\mathcal{K}}$  be the set of atomic concepts and roles in ontology. The inconsistency set of  $I$  for  $\mathcal{K}$ , written as  $ConSet(I, \mathcal{K})$ , is defined as follows:  $ConSet(I, \mathcal{K}) = ConConcepts(I, \mathcal{K}) \cup ConRoles(I, \mathcal{K}) \cup ConEques(I, \mathcal{K})$ , where

- $ConConcepts(I, \mathcal{K}) = \{A(a) \mid A^I(a) = \mathbb{B}, A \in \mathcal{L}_{\mathcal{K}}, a \in \Delta^I\}$ ,
- $ConRoles(I, \mathcal{K}) = \{R(a_1, a_2) \mid R^I(a_1, a_2) = \mathbb{B}, R \in \mathcal{L}_{\mathcal{K}}, a_1, a_2 \in \Delta^I\}$ ,
- $ConEques(I, \mathcal{K}) = \{=(a_1, a_2) \mid (=)^I(a_1, a_2) = \mathbb{B}, a_1, a_2 \in \Delta^I\}$ .

$ConSet(I, \mathcal{K})$  is the set of conflicting atomic assertions in  $\mathcal{K}$ . From Definition 2, we can deduce that a *DL-Lite* ontology  $\mathcal{K}$  is inconsistent if and only if  $ConSet(I, \mathcal{K}) \neq \emptyset$  for every three-valued model  $I$  of  $\mathcal{K}$ .

**Example 3 (Example 1 contd.):** It is easy to check that  $ConSet(I, \mathcal{K}) = \{Employee(a), =(b, c)\}$ .

In Example 1, if we only change the three-valued interpretation of *PhDStud* as  $\langle \{a\}, \{a, b, c\} \rangle$  and obtain another three-valued interpretation  $I'$  of  $\mathcal{K}$ . We can find that  $I'$  is also a three-valued model of  $\mathcal{K}$ . For  $I'$ ,  $ConSet(I', \mathcal{K}) = \{Employee(a), =(b, c), PhDStud(a)\}$ . We can obtain that  $|ConSet(I', \mathcal{K})| > |ConSet(I, \mathcal{K})|$ . That is, given an ontology  $\mathcal{K}$  and a domain, there may exist different models of  $\mathcal{K}$  with different numbers of conflicting atomic individual assertions.

**Definition 3 (Model Ordering):** For a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , let  $I_1, I_2$  be two three-valued models of  $\mathcal{K}$  w.r.t a domain  $\mathcal{D}$ , we say  $I_1$  is preferred to  $I_2$ , written  $I_1 \preceq_{ConSet} I_2$  if and only if  $|ConSet(I_1, \mathcal{K})| \leq |ConSet(I_2, \mathcal{K})|$ .

As usual,  $I_1 \prec_{ConSet} I_2$  denotes  $I_1 \preceq_{ConSet} I_2$  and  $I_2 \not\preceq_{ConSet} I_1$ .  $I_1 \equiv_{ConSet} I_2$  denotes  $I_1 \preceq_{ConSet} I_2$  and  $I_2 \preceq_{ConSet} I_1$ . In the following, we use the model ordering to define preferred models.

**Definition 4:** For a *DL-Lite* ontology  $\mathcal{K}$ , suppose  $3\text{-Model}_{\mathcal{D}}(\mathcal{K})$  is the set of all three-valued models of  $\mathcal{K}$  w.r.t a domain  $\mathcal{D}$ . The set of preferred models of  $\mathcal{K}$  w.r.t domain  $\mathcal{D}$ , written  $preferModel_{\mathcal{D}}(\mathcal{K})$  is defined as follows:  $preferModel_{\mathcal{D}}(\mathcal{K}) = \{I \mid \forall I' \in 3\text{-Model}_{\mathcal{D}}(\mathcal{K}) \text{ implies } I \preceq_{ConSet} I'\}$ .

Calvanese et al. [13] have proposed an important feature about *DL-Lite* ontologies, that is, a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  is satisfiable if and only if  $db(\mathcal{A})$  is a model of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$ . Therefore, we can check whether a *DL-Lite* ontology is satisfiable through checking whether  $db(\mathcal{A})$  is a model of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$ . They have also pointed out that a contradiction on a *DL-Lite<sub>R</sub>* or a *DL-Lite<sub>F</sub>* ontology exists only if a membership assertion in the ABox contradicts a functionality assertion or a NI that is implied by the closure  $cln(\mathcal{T})$ . These results motivate us to define an inconsistency degree by using the domain  $db(\mathcal{A})$  and  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$ . Before we give the definition of inconsistency degree, we would like to show why it is reasonable to consider  $db(\mathcal{A})$  and  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$ . We first give some properties about the relation between  $\langle \mathcal{T}, \mathcal{A} \rangle$  and  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  when they are used to define the inconsistency set.

**Theorem 2:** Let  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  be a *DL-Lite<sub>R</sub>* ontology. Suppose  $\mathcal{K}' = \langle cln(\mathcal{T}), \mathcal{A} \rangle$ ,  $I \in preferModel_{\Delta^{db(\mathcal{A})}}(\mathcal{K}')$  and  $a^I = a$  for each constant  $a \in \Delta^{db(\mathcal{A})}$ . We have  $ConSet(I, \mathcal{K}') \subseteq \mathcal{A}$ .

Based on Theorem 2, we give the following theorem.

**Theorem 3:** Consider a *DL-Lite<sub>R</sub>* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , let  $\mathcal{A}' = \mathcal{A} \setminus ConSet(I, \langle cln(\mathcal{T}), \mathcal{A} \rangle)$ , where  $I \in preferModel_{\Delta^{db(\mathcal{A})}}(\langle cln(\mathcal{T}), \mathcal{A} \rangle)$  and  $a^I = a$  for each

constant  $a \in \Delta^{db(\mathcal{A})}$ . Let  $\mathcal{K}_{repair} = \langle \mathcal{T}, \mathcal{A}' \rangle$ , then  $\mathcal{K}_{repair}$  is satisfiable.

*Proof:* (sketch) First we can give a three-valued model  $I_{temp}$  of  $\langle cln(\mathcal{T}), \mathcal{A}' \rangle$  such that  $ConSet(I_{temp}, \langle cln(\mathcal{T}), \mathcal{A}' \rangle) = 0$ . Assume by contradiction that  $\mathcal{K}_{repair}$  is not satisfiable, then through Theorem 15 in paper [13], we can obtain that  $db(\mathcal{A}')$  is not a model of  $\langle cln(\mathcal{T}), \mathcal{A}' \rangle$  under classical semantics. By construction,  $db(\mathcal{A}')$  cannot contradict a membership assertion in  $\mathcal{A}'$ , so we can deduce that  $db(\mathcal{A}')$  cannot satisfy  $cln(\mathcal{T})$ . In this case, we can prove that  $ConSet(I_{temp}, \langle cln(\mathcal{T}), \mathcal{A}' \rangle) \neq 0$  which contradicts the conclusion of  $ConSet(I_{temp}, \langle cln(\mathcal{T}), \mathcal{A}' \rangle) = 0$ . So the claim holds. ■

From Theorem 3, we know that a  $DL-Lite_{\mathcal{R}}$  ontology  $\mathcal{K}$  will be consistent when we remove from  $\mathcal{K}$  the conflicting assertions of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  obtained from a three-valued preferred model of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  with the domain  $\Delta^{db(\mathcal{A})}$ .

Note that Theorem 2 and Theorem 3 also hold for  $DL-Lite_{\mathcal{F}}$  without functionality assertions. Now we are ready to show an important property that holds for  $DL-Lite_{\mathcal{F}}$  with functionality assertions.

**Theorem 4:** Consider a  $DL-Lite_{\mathcal{F}}$  ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , let  $\mathcal{T}_f$  be the set of functionality assertions in  $\mathcal{T}$  and  $\mathcal{D}$  be a domain where  $|\mathcal{D}| \geq |\Delta^{db(\mathcal{A})}|$ . We have:  $|ConSet(I, \langle \mathcal{T}_f, \mathcal{A} \rangle)| = |ConSet(I', \langle \mathcal{T}_f, \mathcal{A} \rangle)|$ , where  $I \in preferModel_{\Delta^{db(\mathcal{A})}}(\langle \mathcal{T}_f, \mathcal{A} \rangle)$  and  $I' \in preferModel_{\mathcal{D}}(\langle \mathcal{T}_f, \mathcal{A} \rangle)$ .

*Proof:* (sketch) Since  $\langle \mathcal{T}_f, \mathcal{A} \rangle$  only contains functionality assertions, if  $\langle \mathcal{T}_f, \mathcal{A} \rangle$  is inconsistent, we know that there are some membership assertions in  $\mathcal{A}$  that contradict some functionality assertions. So the conflict set is only related with those membership assertions and corresponding functionality assertions. So the claim holds. ■

Theorem 4 tells us that for any three-valued preferred model  $I$  of  $\langle \mathcal{T}_f, \mathcal{A} \rangle$  w.r.t any domain whose cardinality is equal to or greater than that of  $\Delta^{db(\mathcal{A})}$ ,  $|ConSet(I, \langle \mathcal{T}_f, \mathcal{A} \rangle)|$  is a fixed value. Whilst for any domain whose cardinality is less than that of  $\Delta^{db(\mathcal{A})}$ , we cannot find a model of  $\langle \mathcal{T}_f, \mathcal{A} \rangle$ . Based on Definition 9 in [13], we also know that  $\langle \mathcal{T}_f, \mathcal{A} \rangle$  is equal to  $\langle cln(\mathcal{T}_f), \mathcal{A} \rangle$ . So we obtain that the conflicting set of  $|ConSet(I, \langle \mathcal{T}_f, \mathcal{A} \rangle)|$  can be obtained by computing the conflicting set of  $\langle cln(\mathcal{T}_f), \mathcal{A} \rangle$  with the domain  $\Delta^{db(\mathcal{A})}$ .

Theorem 4 only discusses the conflicting set caused by functionality assertions in  $DL-Lite_{\mathcal{F}}$ , we give the relation between the conflicting set caused by NIs and the conflicting set caused functionality assertions as follows.

**Theorem 5:** Given a  $DL-Lite_{\mathcal{F}}$  ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , let  $\mathcal{T}_f$  be the set of functionality assertions in  $\mathcal{T}$  and let  $\mathcal{K}_f = \langle \mathcal{T}_f, \mathcal{A} \rangle$ . Let  $\mathcal{T}_i$  be the set of inclusion assertions in  $\mathcal{T}$  and  $\mathcal{K}_i = \langle \mathcal{T}_i, \mathcal{A} \rangle$ . We have  $|ConSet(I, \langle cln(\mathcal{T}), \mathcal{A} \rangle)| = |ConSet(I_i, \langle cln(\mathcal{T}_i), \mathcal{A} \rangle)| + |ConSet(I_f, \langle cln(\mathcal{T}_f), \mathcal{A} \rangle)|$ , where  $I \in preferModel_{\Delta^{db(\mathcal{A})}}(\langle cln(\mathcal{T}), \mathcal{A} \rangle)$ ,  $I_i \in preferModel_{\Delta^{db(\mathcal{A})}}(\langle cln(\mathcal{T}_i), \mathcal{A} \rangle)$  and

$I_f \in preferModel_{\Delta^{db(\mathcal{A})}}(\langle cln(\mathcal{T}_f), \mathcal{A} \rangle)$ .

In fact,  $\mathcal{K}_i$  in Theorem 5 is the  $DL-Lite_{\mathcal{R}}$  ontology obtained from  $\mathcal{K}$  by removing all functionality assertions in  $\mathcal{T}$ . So  $\mathcal{K}_i$  will be consistent when removing  $ConSet(I_i, \langle cln(\mathcal{T}_i), \mathcal{A} \rangle)$  from  $\mathcal{K}_i$ . By Theorem 5, we know that for a  $DL-Lite_{\mathcal{F}}$  ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , we only need to compute the conflicting set of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  directly instead of considering NIs and functionality assertions separately.

Based on Theorem 3, Theorem 4, Theorem 5, we know that for a  $DL-Lite_{\mathcal{F}}$  ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , there exist two conflicting sets. One set is caused by NIs in  $cln(\mathcal{T})$ ,  $\mathcal{A}$  is consistent with NIs when this set is removed. The other set is caused by functionality assertions in  $cln(\mathcal{T})$  whose cardinality is fixed for any domain and it is easy to check that  $\mathcal{K}$  be a  $DL-Lite_{\mathcal{R}}$  ontology if there is no functionality assertions. Furthermore, based on Theorem 5, we know these two sets can be obtained through computing the conflicting set of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  directly.

Based on the theorems and discussions given in this section, we have the following definition of an inconsistency degree of a  $DL-Lite$  ontology.

**Definition 5 (Inconsistency Degree):** Let  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  be a  $DL-Lite$  ontology, and let  $I$  be a three-valued preferred model of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  w.r.t  $\Delta^{db(\mathcal{A})}$ . The inconsistency degree of  $\mathcal{K}$ , called  $OntoInc(\mathcal{K})$ , is defined as:  $OntoInc(\mathcal{K}) = \frac{|ConSet(I, \langle cln(\mathcal{T}), \mathcal{A} \rangle)|}{|GroundSet(\mathcal{K})|}$ , where  $ConSet(I, \langle cln(\mathcal{T}), \mathcal{A} \rangle)$  is the set of conflicting atomic individual assertions in  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  and  $GroundSet(\mathcal{K})$  is the collection of all possible atomic individual assertions.

**Example 4 (Example 1 contd.):**

We can compute  $cln(\mathcal{T}) = \{PhDStud \sqsubseteq \neg Employee, Stud \sqsubseteq \neg Employee, PhDStud \sqsubseteq \neg Stud, PhDStud \sqsubseteq \neg PhDStud, (func\ hasTutor)\}$ . A three-valued preferred model of  $\langle cln(\mathcal{T}), \mathcal{A} \rangle$  is as follows:  $I_1 = (\Delta^{I_1}, \cdot^{I_1})$ , where  $\Delta^{I_1} = \{a, b, c\}$ ,  $PhDStud^{I_1} = \{\{a\}, \{a, b, c\}\}$ ,  $Stud^{I_1} = \{\{\emptyset\}, \{a, b, c\}\}$ ,  $Employee^{I_1} = \{\{\emptyset\}, \{a, b, c\}\}$ ,  $hasTutor^{I_1} = \{\{(a, b), (a, c)\}, \{(a, a), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c)\}, (=)^{I_1} = \{\{(a, a), (b, b), (c, c), (b, c)\}, \{(b, a), (c, a), (a, b), (c, b), (a, c), (b, c)\}\}$ . For this model,  $GroundSet(\mathcal{K}) = \{PhDStud(a), PhDStud(b), PhDStud(c), Employee(a), Employee(b), Employee(c), Stud(a), Stud(b), Stud(c), hasTutor(a, a), hasTutor(b, a), hasTutor(c, a), hasTutor(a, b), hasTutor(b, b), hasTutor(c, b), hasTutor(a, c), hasTutor(b, c), hasTutor(c, c), = (a, a), = (b, a), = (c, a), = (a, b), = (b, b), = (c, b), = (a, c), = (b, c), = (c, c)\}$ ,  $ConSet(I_1, \langle cln(\mathcal{T}), \mathcal{A} \rangle) = \{PhDStud(a), = (b, c)\}$ , so  $OntoInc(\mathcal{K}) = \frac{2}{27}$ .

From Definition 5, we can show the following properties for  $DL-Lite$  ontologies.

**Proposition 6:** Let  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ ,  $\mathcal{K}_1, \mathcal{K}_2$  are  $DL-Lite$  ontologies, we have the following properties:

- (R1)  $OntoInc(\mathcal{K})=0$  when  $\mathcal{K}$  is consistent.
- (R2)  $OntoInc(\mathcal{K}_1) \leq OntoInc(\mathcal{K}_2)$  when  $GroundSet(\mathcal{K}_1) = GroundSet(\mathcal{K}_2)$  and  $\mathcal{K}_1 \subseteq \mathcal{K}_2$ .

(R3)  $\text{OntoInc}(\mathcal{K}) \leq \text{OntoInc}(\mathcal{K} \cup S)$ , where  $S$  is a set of membership assertions which are not in  $\mathcal{K}$  but in  $\text{GroundSet}(\mathcal{K})$ .

In Proposition 6, R2 says that the inconsistency degree of an ontology will not decrease if we add to it new axioms which do not change its ground set. R3 is a special case of R2. It says that the inconsistency degree of an ontology will not decrease if we add to it more membership assertions in its ground set.

## V. ALGORITHM FOR COMPUTING INCONSISTENCY DEGREE

In this section, we give an algorithm to compute a preferred model  $I$  of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$  with the domain  $\Delta^{\text{db}(\mathcal{A})}$ . First, we need to extend some definitions in [13]. We will use the symbol “\*” to denote all constants in the domain  $\Delta$ . For example, assume a domain  $\Delta = \{a, b, c\}$ , then  $R(a, *)$  denotes the set  $\{R(a, a), R(a, b), R(a, c)\}$ .

We start with defining *applicable negative inclusions (NIs)* and *applicable functionality assertions (FunAss)*, then we use applicable NIs and FunAss to construct a chase for  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ , written *chase-cln*( $\mathcal{K}$ ). With the notion *chase-cln*( $\mathcal{K}$ ) in place, we give an algorithm to compute a preferred model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ .

**Definition 6 (Applicable Negative Inclusions):** For a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , let  $\mathcal{S}_p = \mathcal{A}$  and  $x$  be a constant in  $\Delta^{\text{db}(\mathcal{A})}$ . Suppose  $\mathcal{S}$  is a certain set of membership assertions. A NI  $\alpha \in \text{cln}(\mathcal{T})$  is applicable in  $\mathcal{S}$  to a membership assertion  $f \in \mathcal{S}_p$  if

- $\alpha = A_1 \sqsubseteq \neg A_2, f = A_1(a)$  and  $\neg A_2(a) \notin \mathcal{S}$ ;
- $\alpha = \exists R \sqsubseteq \neg A, f = R(a, b)$  and  $\neg A(a) \notin \mathcal{S}$ ;
- $\alpha = \exists R^- \sqsubseteq \neg A, f = R(a, b)$  and  $\neg A(b) \notin \mathcal{S}$ ;
- $\alpha = R_1 \sqsubseteq \neg R_2, f = R_1(a, b)$  and  $\neg R_2(a, b) \notin \mathcal{S}$ ;
- $\alpha = R_1 \sqsubseteq \neg R_2^-, f = R_1(a, b)$  and  $\neg R_2(b, a) \notin \mathcal{S}$ ;
- $\alpha = R_1^- \sqsubseteq \neg R_2, f = R_1(a, b)$  and  $\neg R_2(b, a) \notin \mathcal{S}$ ;
- $\alpha = R_1^- \sqsubseteq \neg R_2^-, f = R_1(a, b)$  and  $\neg R_2(a, b) \notin \mathcal{S}$ ;
- $\alpha = A \sqsubseteq \neg \exists R, f = A(a)$  and  $\exists x, \neg R(a, x) \notin \mathcal{S}$ ;
- $\alpha = A \sqsubseteq \neg \exists R^-, f = A(a)$  and  $\exists x, \neg R(x, a) \notin \mathcal{S}$ ;
- $\alpha = \exists R_1 \sqsubseteq \neg \exists R_2, f = R_1(a, b)$  and  $\exists x, \neg R_2(a, x) \notin \mathcal{S}$ ;
- $\alpha = \exists R_1^- \sqsubseteq \neg \exists R_2^-, f = R_1(a, b)$  and  $\exists x, \neg R_2(x, b) \notin \mathcal{S}$ ;
- $\alpha = \exists R_1 \sqsubseteq \neg \exists R_2^-, f = R_1(a, b)$  and  $\exists x, \neg R_2(x, a) \notin \mathcal{S}$ ;
- $\alpha = \exists R_1^- \sqsubseteq \neg \exists R_2, f = R_1(a, b)$  and  $\exists x, \neg R_2(b, x) \notin \mathcal{S}$ .

**Definition 7 (Applicable Functionality assertion):**

For a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , let  $\mathcal{S}_p = \mathcal{A}$  and  $x$  be a constant in  $\Delta^{\text{db}(\mathcal{A})}$ . Suppose  $\mathcal{S}$  is a certain set of membership assertions. A FunAss  $\alpha \in \text{cln}(\mathcal{T})$  is applicable in  $\mathcal{S}$  to a membership assertion  $f \in \mathcal{S}_p$  if

- $\alpha = (\text{funct } R), f = R(a, b)$  and  $\exists x, R(a, x) \in \mathcal{S}_p, \neq(b, x) \notin \mathcal{S}$ ;
- $\alpha = (\text{funct } R^-), f = R(a, b)$  and  $\exists x, R(x, b) \in \mathcal{S}_p, \neq(a, x) \notin \mathcal{S}$ .

Applicable NIs and FunAss can be used to construct *chase-cln*( $\mathcal{K}$ ). Roughly speaking, *chase-cln*( $\mathcal{K}$ ) is a set assertion constructed step-by-step from  $\mathcal{A}$  and  $\mathcal{S}$ . At each step

of construction, a NI or a functionality assertion  $\alpha \in \text{cln}(\mathcal{T})$  is applied to a membership assertion  $f \in \mathcal{S}_p$ , so that a new suitable membership assertion is added to  $\mathcal{S}$ , thus obtaining a new set  $\mathcal{S}'$  in which  $\alpha$  is no longer applicable to  $f$ . For example, if  $\alpha = A_1 \sqsubseteq \neg A_2$  is applicable to  $f = A_1(a)$ , the membership assertion to be added to  $\mathcal{S}$  is  $\neg A_2(a)$ , that is,  $\mathcal{S}' = \mathcal{S} \cup \neg A_2(a)$ . Our construction process about the *chase-cln*( $\mathcal{K}$ ) is precisely given below.

**Definition 8:** Given a *DL-Lite* KB  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ ,  $\text{cln}(\mathcal{T})$  is the closure of  $\mathcal{T}$ . We construct a sequence of sets  $\mathcal{S}_j$  inductively as follows:

$\mathcal{S}_p = \mathcal{A}, \mathcal{S}_0 = \emptyset. \mathcal{S}_{j+1} = \mathcal{S}_j \cup f_{\text{new}}$ , where  $f_{\text{new}}$  is a membership assertion obtained as follows: Let  $f$  be a membership assertion in  $\mathcal{S}_p$  such that there exists an axiom  $\alpha \in \text{cln}(\mathcal{T})$  applicable in  $\mathcal{S}_j$  to  $f$ .

**Case  $\alpha, f$  of**

- (cr1)  $\alpha = A_1 \sqsubseteq \neg A_2$  and  $f = A_1(a)$  **then**  $f_{\text{new}} = \neg A_2(a)$
- (cr2)  $\alpha = \exists R \sqsubseteq \neg A$  and  $f = R(a, b)$  **then**  $f_{\text{new}} = \neg A(a)$
- (cr3)  $\alpha = \exists R^- \sqsubseteq \neg A$  and  $f = R(a, b)$  **then**  $f_{\text{new}} = \neg A(b)$
- (cr4)  $\alpha = R_1 \sqsubseteq \neg R_2$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(a, b)$
- (cr5)  $\alpha = R_1 \sqsubseteq \neg R_2^-$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(b, a)$
- (cr6)  $\alpha = R_1^- \sqsubseteq \neg R_2$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(b, a)$
- (cr7)  $\alpha = R_1^- \sqsubseteq \neg R_2^-$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(a, b)$
- (cr8)  $\alpha = A \sqsubseteq \neg \exists R$  and  $f = A(a)$   
**then**  $f_{\text{new}} = \neg R(a, *)$
- (cr9)  $\alpha = A \sqsubseteq \neg \exists R^-$  and  $f = A(a)$   
**then**  $f_{\text{new}} = \neg R(*, a)$
- (cr10)  $\alpha = \exists R_1 \sqsubseteq \neg \exists R_2$  and  $f = R_1(a, b)$   
**then then**  $f_{\text{new}} = \neg R_2(a, *)$
- (cr11)  $\alpha = \exists R_1^- \sqsubseteq \neg \exists R_2^-$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(*, b)$
- (cr12)  $\alpha = \exists R_1 \sqsubseteq \neg \exists R_2^-$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(*, a)$
- (cr13)  $\alpha = \exists R_1^- \sqsubseteq \neg \exists R_2$  and  $f = R_1(a, b)$   
**then**  $f_{\text{new}} = \neg R_2(b, *)$
- (cr14)  $\alpha = (\text{funct } R)$  and  $f = R(a, b), \forall x, R(a, x) \in \mathcal{S}_p$   
**then**  $f_{\text{new}} = (\neq(b, x))$
- (cr15)  $\alpha = (\text{funct } R^-)$  and  $f = R(a, b), \forall x, R(x, b) \in \mathcal{S}_p$   
**then**  $f_{\text{new}} = (\neq(x, a))$ .

In Definition 8, we know that the number of NIs and FunAss in  $\text{cln}(\mathcal{T})$  is fixed and the number of membership assertions in  $\mathcal{S}_p$  is also fixed because  $\mathcal{S}_p = \mathcal{A}$ . Furthermore, a NI or a FunAss in  $\text{cln}(\mathcal{T})$  can be applied at most once to a membership assertion in  $\mathcal{S}_p$  (afterwards, the precondition is not satisfied and the NI or FunAss is no longer applicable), and a rule can be applied at most  $m$  times to some membership assertions, where  $m$  is the number of NIs and FunAss in  $\text{cln}(\mathcal{T})$ . We also know that no new constant is produced in the construction process. So the set of membership assertions obtained, written  $\mathcal{S}_n$ , is the finite union of all  $\mathcal{S}_j$ , namely,

$$\mathcal{S}_n = \bigcup_{j \in N} \mathcal{S}_j.$$

Let  $\text{chase-cln}(\mathcal{K}) = \mathcal{S}_p \cup \mathcal{S}_n$ , where  $\mathcal{S}_p$  equals to  $\mathcal{A}$ ,  $\mathcal{S}_n$  is composed by membership assertions which are obtained through applying NIs or FunAss to membership assertions in  $\mathcal{S}_p$ . Note that  $\mathcal{S}_p$  and  $\mathcal{S}_n$  are both finite, so  $\text{chase-cln}(\mathcal{K})$  is finite. Furthermore,  $\text{chase-cln}(\mathcal{K})$  is unique because  $\text{cln}(\mathcal{T})$  and  $\mathcal{S}_p$  have not been changed in the construction process, and the construction of  $\text{chase-cln}(\mathcal{K})$  only depends on  $\text{cln}(\mathcal{T})$ ,  $\mathcal{S}_p$  and all constants occurring in  $\mathcal{A}$  (uses for obtaining membership assertions of the form  $\neg R(*, a)$  or  $\neg R(a, *)$ ).

**Example 5 (Example 1 contd.):** From  $\mathcal{K}$ , we can obtain that  $\Delta^{db(\mathcal{A})} = \{a, b, c\}$  and  $\text{cln}(\mathcal{T}) = \{\text{PhDStud} \sqsubseteq \neg \text{Employee}, \text{Stud} \sqsubseteq \neg \text{Employee}, \text{PhDStud} \sqsubseteq \neg \text{PhDStud}, \text{PhDStud} \sqsubseteq \neg \text{Stud}, (\text{funct } \text{hasTutor})\}$ .  $\mathcal{S}_p = \{\text{PhDStud}(a), \text{hasTutor}(a, b), \text{hasTutor}(a, c)\}$ ,  $\mathcal{S}_0 = \emptyset$ .

- For  $\text{PhDStud} \sqsubseteq \neg \text{PhDStud}$ , because  $\text{PhDStud}(a) \in \mathcal{S}_p$  and  $\neg \text{PhDStud}(a) \notin \mathcal{S}_0$ , these would trigger the chase rule **cr1**, so  $\neg \text{PhDStud}(a)$  will be added to  $\mathcal{S}_0$  obtaining  $\mathcal{S}_1 = \{\neg \text{PhDStud}(a)\}$ ; In an analogous way,  $\neg \text{Employee}(a)$  will be added to  $\mathcal{S}_1$  obtaining  $\mathcal{S}_2$  and  $\neg \text{Stud}(a)$  will be added to  $\mathcal{S}_2$  obtaining  $\mathcal{S}_3$ , that is  $\mathcal{S}_3 = \{\text{PhDStud}(a), \text{Employee}(a), \neg \text{Stud}(a)\}$ ;
- For  $(\text{funct } \text{hasTutor})$ , because  $\text{hasTutor}(a, b) \in \mathcal{S}_p$  and  $\text{hasTutor}(a, c) \in \mathcal{S}_p$ ,  $\text{=(}b, c) \notin \mathcal{S}_3$ , these would trigger the chase rule **cr14**, so  $\text{=(}b, c)$  will be added to  $\mathcal{S}_3$  obtaining  $\mathcal{S}_4 = \{\text{PhDStud}(a), \text{Employee}(a), \neg \text{Stud}(a), \text{=(}b, c)\}$ ;

So  $\text{Chase-cln}(\mathcal{K}) = \mathcal{S}_p \cup \mathcal{S}_n$ , where  $\mathcal{S}_p = \{\text{PhDStud}(a), \text{hasTutor}(a, b), \text{hasTutor}(a, c)\}$ ,  $\mathcal{S}_n = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3 = \{\neg \text{PhDStud}(a), \neg \text{Employee}(a), \neg \text{Stud}(a), \text{=(}b, c)\}$ .

In this paper, inclusion axioms are interpreted as *internal inclusions* which propagate contradictory information forwardly, but not backwardly as it does not allow for contraposition reasoning (see [8] for definition of internal inclusion). Indeed,  $\mathcal{S}_n$  is a result of propagating contradictory information based on internal inclusion through *ABox*. This is because in the process of constructing  $\mathcal{S}_n$ , only negative inclusions and function assertions are considered, regardless of positive inclusions. Algorithm 1 gives an algorithm for computing a three-valued interpretation  $I_{cln}$  for  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ , written as  $I_{cln} = \langle \Delta^{I_{cln}}, \cdot^{I_{cln}} \rangle$ , based on  $\text{chase-cln}(\mathcal{K})$ , i.e.,  $\mathcal{S}_p$  and  $\mathcal{S}_n$ . The returned value  $I_{cln}$  of Algorithm 1 is a preferred three-valued model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$  (see Theorem 7).

**Example 6 (Example 5 continue):** By Algorithm 1, we can obtain  $I_{cln} = \langle \Delta^{I_{cln}}, \cdot^{I_{cln}} \rangle$ , where  $\Delta^{I_{cln}} = \{a, b, c\}$ ,  $\text{PhDStud}^{I_{cln}} = \langle \{a\}, \{a, b, c\} \rangle$ ,  $\text{Employee}^{I_{cln}} = \langle \emptyset, \{a, b, c\} \rangle$ ,  $\text{Stud}^{I_{cln}} = \langle \emptyset, \{a, b, c\} \rangle$ ,  $\text{hasTutor}^{I_{cln}} = \langle \{(a, b), (a, c)\}, \{(a, a), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c)\} \rangle$ ,  $\text{=(}^{I_{cln}} = \langle \{(a, a), (b, b), (c, c), (b, c)\}, \{(a, b), (a, c), (b, a), (b, c), (c, a), (c, b)\} \rangle$ .

The following theorem shows the correctness of our algorithm.

---

**Algorithm 1** Algorithm for computing  $I_{cln}$ 


---

- 1: Input:  $\mathcal{S}_p, \mathcal{S}_n$
  - 2: Output:  $I_{cln}$
  - 3:  $\Delta^{I_{cln}} = \Delta^{db(\mathcal{A})}$
  - 4: **for** each constant  $x$  occurring in  $\text{Chase-cln}(\mathcal{K})$  **do**
  - 5:    $x^{I_{cln}} = x$
  - 6: **end for**
  - 7: **for** each atomic concept  $A$  **do**
  - 8:    $A^{I_{cln}} = \langle \{a \mid A(a) \in \mathcal{S}_p\}, \{b \mid \neg A(b) \in \mathcal{S}_n\} \cup (\Delta^{db(\mathcal{A})} \setminus \{a \mid A(a) \in \mathcal{S}_p\}) \rangle$
  - 9: **end for**
  - 10: **for** each atomic role  $P$  **do**
  - 11:    $P^{I_{cln}} = \langle \{(a_1, a_2) \mid P(a_1, a_2) \in \mathcal{S}_p\}, \{(b_1, b_2) \mid \neg P(b_1, b_2) \in \mathcal{S}_n\} \cup ((\Delta^{db(\mathcal{A})} \times \Delta^{db(\mathcal{A})}) \setminus \{(a_1, a_2) \mid P(a_1, a_2) \in \mathcal{S}_p\}) \rangle$
  - 12: **end for**
  - 13:  $\text{=(}^{I_{cln}} = \langle \{(a_1, a_2) \mid (a_1 = a_2) \in \mathcal{S}_n\} \cup \{(a, a) \mid \forall a \in \Delta^{db(\mathcal{A})}, \{(b_1, b_2) \mid \forall b_1, b_2 \in \Delta^{db(\mathcal{A})} \text{ and } b_1 \neq b_2\} \rangle$ .
- 

**Theorem 7:** Let  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$  be a *DL-Lite* ontology, then the three-valued interpretation  $I_{cln}$  obtained by Algorithm 1 is a preferred three-valued model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$  with the domain  $\Delta^{db(\mathcal{A})}$ .

*Proof:* (sketch) We first show that  $I_{cln}$  is a model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ . Because  $\mathcal{A} \subseteq \text{chase-cln}(\mathcal{K})$ , so  $I_{cln}$  satisfies all membership assertions in  $\mathcal{A}$ . Then, we only need to prove that  $I_{cln} \models \text{cln}(\mathcal{T})$ . Suppose by contradiction that a NI of the form  $A_1 \sqsubseteq \neg A_2 \in \text{cln}(\mathcal{T})$ , where  $A_1$  and  $A_2$  are atomic concepts, is not satisfied by  $I_{cln}$ . Then there exists a constant  $a \in \Delta^{db(\mathcal{A})}$  such that  $A_1(a) \in \mathcal{S}_p$  and  $\neg A_2(a) \notin \mathcal{S}_n$ . However, such a situation would trigger the rule **cr1**, thus causing the adding of  $\neg A_2(a)$  in  $\mathcal{S}_n$  at some step, hence contradicting the assumption. NIs of other form can be proved in an analogous way.

Second we show that  $I_{cln}$  is a preferred model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ . Suppose  $I_{cln}$  is not a preferred model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ . So there must exist a preferred model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ , named  $I$ , and an atomic assertion  $\alpha$  such that  $\alpha^{I_{cln}} = \mathbb{B}$ ,  $\alpha^I \neq \mathbb{B}$ . Suppose  $\alpha = A(a)$ . By Definition 8 and Algorithm 1, there must exist the following cases:  $\{C \sqsubseteq \neg A, C(a), A(a)\}$  or  $\{\exists R \sqsubseteq \neg A, R(a, b), A(a)\}$  or  $\{\exists R^- \sqsubseteq \neg A, R(b, a), A(a)\}$  in  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ . Because  $A^I(a) \neq \mathbb{B}$ , so  $A^I(a) = t$ . According to the three-valued semantics of internal inclusion,  $I$  can not satisfy either case above. That is,  $I$  is not a three-valued model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ , contradictive with the assumption. Similarly, we can get the same result for the cases that  $\alpha$  is  $R(a, b)$  or  $\text{=(} (a, b)$ . So  $I_{cln}$  is a preferred model of  $\langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle$ . ■

**Example 7 (Example 6 continue):**  $|\text{GroundSet}(\mathcal{K})| = 27$ ,  $\text{ConSet}(I_{cln}, \langle \text{cln}(\mathcal{T}), \mathcal{A} \rangle) = \{\text{PhDStud}(a), \text{=(} (b, c)\}$ , so  $\text{OntoInc}(\mathcal{K}) = \frac{2}{27}$ .

We consider the complexity of computing the inconsis-

tency degree as follows.

**Theorem 8:** Given a *DL-Lite* ontology  $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ , the inconsistency degree of  $\mathcal{K}$  can be computed in PTime in the size of  $\mathcal{K}$ .

*Proof:* (sketch) First  $cln(\mathcal{T})$  is polynomially related to the size of TBox  $\mathcal{T}$  [13]. By Definition 8, the time of computing  $chase-cln(\mathcal{K})$  is at most  $|\mathcal{A}| \times |cln(\mathcal{T})| \times |\Delta^{db(\mathcal{A})}|$  because  $|\mathcal{S}_p| = |\mathcal{A}|$  and  $|\mathcal{S}_n| = |\mathcal{A}| \times |cln(\mathcal{T})| \times |\Delta^{db(\mathcal{A})}|$  in the worst case. That is,  $|\mathcal{S}_n|$  is polynomial to the size of  $\mathcal{K}$ . By Algorithm 1, we know that the time of computing model  $I_{cln}$  is  $c \times (|\mathcal{S}_p| + |\mathcal{S}_n| + |\Delta^{db(\mathcal{A})}|) + r \times (|\mathcal{S}_p| + |\mathcal{S}_n| + |\Delta^{db(\mathcal{A})}|)$ , where  $c$  and  $r$  denote the number of concepts and roles in  $\mathcal{K}$  respectively. So  $I_{cln}$  is obtained in polynomial time to the size of  $\mathcal{K}$ . From  $I_{cln}$ , the inconsistency degree can be computed in polynomial time to  $|\Delta^{I_{cln}}|$ , which is polynomially related to the size of  $\mathcal{K}$ . ■

## VI. CONCLUSION AND FUTURE WORK

Measuring inconsistency in inconsistent ontologies is an important problem in ontology engineering. In this paper, we considered the problem of computing inconsistency degree for *DL-Lite* ontologies. We first gave a three-valued semantics for *DL-Lite*. Then we defined an inconsistency degree of *DL-Lite* ontologies. The inconsistency degree is only a single value, not a sequence. Arguably, this single value is easier to be used to deal with inconsistencies than a sequence of values given in [5]. Furthermore, we also proposed a polynomial-time algorithm to compute the inconsistency degree of an ontology. As a future work, we will implement our algorithm and provide experimental results.

## ACKNOWLEDGMENT

We are participating in a project of the national grand fundamental research 973 project of China under Grant (No.2007CB307100, No.2007CB307106), which is named “the Fundamental Research on the Architecture of Universal Trustworthy Network and Pervasive Services”. This project has implemented the unified description ontology for the data and services in the Universal Network. The inconsistency measurements played an important role in the creation of the unified description ontology. Our research is also supported by the Specialized Research Foundation of Doctoral Program of Higher Education of China under Grant (No.20050004008).

## REFERENCES

- [1] G. Qi, P. Haase, Z. Huang, Q. Ji, J. Z. Pan, and J. Völker. A kernel revision operator for terminologies - algorithms and evaluation. In A. P. Sheth et al., editor, *Proceedings of the 7th International Semantic Web Conference*, pages 419–434, 2008.
- [2] P. Haase, F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y. Sure. A framework for handling inconsistency in changing ontologies. In E. Motta Y. Gil, V. Richard Benjamins, and M. A. Musen, editors, *Proceedings of the 2nd European Semantic Web Conference*, pages 353–367, 2005.
- [3] A. Hunter and S. Konieczny. Measuring inconsistency through minimal inconsistent sets. In *Proceedings of 11th International Conference on Principles of Knowledge Representation and Reasoning*, pages 358–366, 2008.
- [4] G. Qi and A. Hunter. Measuring incoherence in description logic-based ontologies. In K. Aberer et al., editor, *Proceedings of the 6th International Semantic Web Conference*, pages 381–394, 2007.
- [5] Y. Ma, G. Qi, P. Hitzler, and Z. Lin. Measuring inconsistency for description logics based on paraconsistent semantics. In K. Mellouli, editor, *Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 30–41, 2007.
- [6] A. Hunter and S. Konieczny. Approaches to measuring inconsistent information. In L. E. Bertossi, A. Hunter, and T. Schaub, editors, *Inconsistency Tolerance*, volume 3300, pages 191–236, 2005.
- [7] X. Deng, V. Haarslev, and N. Shiri. Measuring inconsistencies in ontologies. In E. Franconi, M. Kifer, and W. May, editors, *Proceedings of the 4th European Semantic Web Conference*, pages 326–340, 2007.
- [8] Y. Ma, P. Hitzler, and Z. Lin. Algorithms for paraconsistent reasoning with OWL. In E. Franconi, M. Kifer, and W. May, editors, *Proceedings of the 4th European Semantic Web Conference*, pages 399–413, 2007.
- [9] A. Hunter. How to act on inconsistent news: Ignore, resolve, or reject. *Data Knowl. Eng.*, 57(3):221–239, 2006.
- [10] F. Baader, R. Peñalosa, and B. Suntisrivaraporn. Pinpointing in the description logic  $EL^+$ . In *Proc. of KI'07*, pages 52–67, 2007.
- [11] Y. Ma, G. Qi, P. Hitzler, and Z. Lin. An algorithm for computing inconsistency measurement by paraconsistent semantics. In K. Mellouli, editor, *Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 91–102, 2007.
- [12] Domenico Lembo and Marco Ruzzi. Consistent query answering over description logic ontologies. In M. Marchiori, J. Z. Pan, and C. de Sainte Marie, editors, *Proceedings of the First International Conference on Web Reasoning and Rule Systems*, pages 194–208, 2007.
- [13] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.
- [14] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. Description logic terminology. In F. Baader et al., editor, *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
- [15] Y. Ma, P. Hitzler, and Z. Lin. Paraconsistent reasoning for expressive and tractable description logics. In F. Baader, C. Lutz, and B. Motik, editors, *Description Logics*, 2008.