J Intell Inf Syst (2013) 40:375–403
DOI 10.1007/s10844-012-0230-7

An argumentation framework for description logic ontology reasoning and management

Xiaowang Zhang · Zuoquan Lin

Received: 29 May 2012 / Revised: 15 November 2012 / Accepted: 15 November 2012 /
Published online: 8 December 2012

Abstract This paper presents an argumentation framework for reasoning and management in (inconsistent or incoherent) description logic ontologies which contain conflicts. First, a new argumentation framework obtained by combining Besnard and Hunter's framework with binary argumentation is introduced to frame the inner relation over axioms in an ontology. A dialogue mechanism, based on this framework, is then presented to derive meaningful consequences from inconsistent ontologies. Three novel operators are developed to repair those axioms or assertions which cause inconsistency or incoherency of ontologies by using this framework. Within this framework, an inconsistency is neither directly assigned a contradictory value nor roughly removed but further analyzed and evaluated. Because of this, reasoning within it satisfies some important logical properties such as consistency-preserving and justifiability. Moreover, it provides an alternative scenario for maintaining consistency and coherency of ontologies with giving consideration to both semantics and syntax. Thus the repaired results by using the proposed framework not only keep the closer semantics but also preserve the syntactic structure of original ontologies.

Keywords Ontology · Description logic · Handling inconsistency · Argumentation · Ontology management · Paraconsistent reasoning

X. Zhang (✉)
Hasselt University and Transnational University of Limburg, Diepenbeek, Belgium
e-mail: xiaowang.zhang@uhasselt.be

Z. Lin
School of Mathematical Sciences, Peking University, Beijing, P. R. China

Present Address:
Z. Lin
Peking University, No. 5 Yiheyuan Road, Haidian District, 100871 Beijing, China

1 Introduction

In computer science, an ontology is a formal representation of the knowledge about (some aspect of) the world by a set of concepts within a domain and the relationships between those concepts; it introduces vocabulary describing various aspects of the domain being modeled, and provides an explicit specification of the intended meaning of the vocabulary (Horrocks 2008). Ontologies are widely used to represent knowledge in the Semantic Web (Berners-Lee et al. 2001), which is conceived as a future generation of the World Wide Web (WWW) by defining the meaning (semantics) of information and services on the Web in order to make it possible for the computer programs to process and use the Web content (Berners-Lee et al. 2001). The Web Ontology Language (OWL) is proved to be a highly successful class of knowledge representation languages to represent ontologies in the Semantic Web (Berners-Lee et al. 2001). Description logics (DLs), as fragments of *first-order logic* (FOL), are the logical foundation of OWL (Baader et al. 2003). Compared with *propositional logic* (PL), there are more expressive decision problems which can be solved by many DL reasoning engines/algorithms. However, we have to live with the fact that ontologies in a real world might be rarely perfect since the Semantic Web is an open, constantly changing and collaborative environment. Many reasons, such as modeling errors, migration from other formalisms, knowledge integration, knowledge merging and knowledge evolution, potentially bring ontologies with conflicts (Meyer et al. 2005; Bertossi et al. 2005; Kalyanpur et al. 2006a; Odintsov and Wansing 2008). For example, during the recent development of an OWL ontology at NASA's Jet Propulsion Laboratory, the class "*OceanCrustLayer*" was found to be inconsistent (Horrocks 2008). Engineers discovered (with the help of debugging tools) that the class was defined as both a region and a layer, one of which (layer) was a 2-dimensional object and the other a 3-dimensional object. The inconsistency thus highlighted a fundamental error in the ontology's design. Then it is unrealistic to expect that real ontologies are always logically consistent. Unfortunately, DLs break down in the presence of contradictory knowledge in ontologies.

As a result, the issue of handling DL ontologies with conflicts has attracted much attention in the Semantic Web community.

This issue has been researched from two main directions: the first, based on the assumption that inconsistency is treated as a natural phenomenon in realistic data, is applying a non-standard inference to avoid the explosive entailment and the second, based on the assumption that inconsistent information indicates erroneous data, is removing those axioms which cause conflicts from ontologies to maintain consistency and coherency of ontologies. The first is a topic of inconsistency-tolerant reasoning (Odintsov and Wansing 2008) and the second is a key task of ontology management (Davies et al. 2009).

There are some existing approaches (Schlobach and Cornet 2003; Huang et al. 2005; Qi and Du 2009; Patel-Schneider 1989; Ma et al. 2007; Odintsov and Wansing 2008; Zhang et al. 2009, 2010; Zhang and Lin 2012) to avoid the explosive entailment in reasoning with inconsistent ontologies. The main idea of those approaches is to make the inference rule "*ex falso quodlibet*: $\{\bot(a)\} \vdash \phi$ or $\{\top \sqsubseteq \bot\} \models \phi$ where a is an arbitrary individual and ϕ is an arbitrary axiom" invalid to obtain some meaningful information from an inconsistent ontology. They are also based on two fundamentally different strategies. One is forbidding some axioms which possibly cause inconsistency to be used in reasoning with sub-ontologies of an ontology

(Schlobach and Cornet 2003; Huang et al. 2005; Qi and Du 2009). However, based on this strategy, we might lose some potentially meaningful consequents from ontologies since sub-ontologies are hard to capture the whole ontology. The other is introducing non-classical semantics to provide non-standard inferences to tolerate inconsistency (Patel-Schneider 1989; Ma et al. 2007; Odintsov and Wansing 2008; Zhang et al. 2009, 2010; Zhang and Lin 2012). The main idea of current proposals is using multi-valued inference. However, the inference power of them is much weaker than that of the standard inference, even for consistent ontologies. For instance, four-valued entailment (an inclusion relation between four-valued models of a set of axioms and four-valued models of an axiom) (Patel-Schneider 1989; Ma et al. 2007; Odintsov and Wansing 2008) and three-valued entailment (an inclusion relation between three-valued models of a set of axioms and three-valued models of an axiom) (Zhang et al. 2010) do not satisfy all three basic inference rules, namely, *modus ponens*: $\{C(a), C \sqsubseteq D\} \models D(a)$, *disjunctive syllogism*: $\{\neg C(a), C \sqcup D\} \models D(a)$ and *modus tollens*: $\{\neg D(a), C \sqsubseteq D\} \models \neg C(a)$. Quasi-classical entailment (an inclusion relation between strong models of a set of axioms and weak models of an axiom) (Zhang et al. 2009; Zhang and Lin 2012) do not satisfy the *law of excluded middle*: $\mathcal{O} \models \top(a)$ (where $\mathcal{O}$ is an arbitrary ontology). Indeed, they have a common weakness that inconsistencies are not further analyzed in-depth but either isolated even being discarded to maintain maximal consistency or ignored by evaluating a contradictory value "B" (i.e., both *true* and *false*). In this sense, they look rough in treating inconsistencies.

On the other hand, there are some approaches to eliminate conflicts to maintain consistency and coherency of DL ontologies (Schlobach 2005; Meyer et al. 2006; Kalyanpur et al. 2006b; Du and Shen 2008; Qi and Du 2009; Ji et al. 2009; Wang et al. 2010). Those traditional approaches to maintaining coherency and consistency of DL ontologies are almost based on syntactical modification (Schlobach 2005; Meyer et al. 2006; Kalyanpur et al. 2006b; Du and Shen 2008). Though these syntax-based approaches obey the principle of minimal modification in syntax, they might hardly preserve the semantics of original ontologies. Recently, some model-based approaches are proposed to eliminate conflicts between two DL ontologies (where one (called the old) is to be revised by the other (called the new) (Qi and Du 2009; Ji et al. 2009; Wang et al. 2010). The main idea of them is to select those models, which are the closest to models of the new, from models of the old as the candidate models of revised ontologies which are obtained by using the new to revise the old. However, a revision problem always involves two ontologies, which is not the case for inconsistency repair (resolution) in general.

To address this problem, in this paper, we will propose a novel approach based on *argumentation framework* to reason with and manage DL ontologies. The first version of argumentation framework is proposed by Dung (1995b) (we call Dung's framework) in order to provide a proof-theoretic semantics for nonmonotonic logic so that some faithful conclusions could be inferred from knowledge bases with conflicts by applying the dialogue mechanism. Indeed, Dung's framework is in a graph-based frame where each node is an argument and the set of edges describes an "*attack*" relation between arguments. Based on Dung's framework, Besnard and Hunter (2001) (we abbreviate the framework proposed by them to BH's framework) presented a tree-based argumentation framework for propositional knowledge bases with preserving the terminability of arguing processes. In this sense, BH's argumentation framework could be taken as an instantiation of Dung's framework.

As an alternative scenario, based on BH's framework, this paper presents an argumentation framework to manage and reason with DL ontologies with conflicts. However, it is not straightforward to generalize BH's framework to DLs because the attack relation over DL ontologies should capture not only inconsistency but also incoherency which is not viewed as a conflict in PL. Moreover, though this tree-based framework for PL can be feasibly obtained, for DL ontologies, it might be not always constructed since a DL ontology can have infinite number of models and some model can also be infinite. Additionally, the deduction relation in defining arguments can no longer work since the DL syntactic constructors connect not axioms but DL concepts and roles.

This paper presents an argumentation framework for $\mathcal{ALC}$ ontologies and, within this framework, some novel approaches are developed to reason with and manage inconsistent or incoherent ontologies. This paper is a revised and extended version of previous proceedings of DL 2009 and CAAI 2010. We select $\mathcal{ALC}$ because $\mathcal{ALC}$ is a basic member of DL family. The main innovations and contributions of this paper can be summarized as follows:

- Introducing an argumentation framework for DL ontologies by modifying BH's argumentation to capture kinds of conflicts, namely incoherencies and inconsistencies. Firstly, we introduce arguments, undercuts, conservative relation over undercuts. Then argument trees for axioms are built in canonical undercuts by using some expansion rules. Finally, this framework for an axiom is a pair of argument trees for and against the axiom.
- Presenting an argumentative entailment relationship between an ontology and an axiom within argumentation framework. We can show that the argumentative entailment is not only non-explosive but also satisfying some important logical properties such as justifiability, consistency-preserving, nonmonotonicity and splitting property.
- Developing three argumentative operators (a normal operator and the other two approximative operators) to maintain coherency and consistency of $\mathcal{ALC}$ ontologies. We show that the argumentative operator could maintain consistency of ontologies by removing all redundant axioms which could not be justified from original ontologies. Such a scenario satisfies a property of justifiability, that is, each axiom of a repaired ontology could be self-protected. Additionally, such two approximative operators enrich the repaired ontologies by adding potentially justifiable axioms. For some users, the approximative repairs are suitable for application in providing them with many choices.

The rest of this paper is organized as follows. Section 2 reviews briefly the syntax and semantics of $\mathcal{ALC}$. Section 3 defines argumentation framework for ontologies. Section 4 discusses using argumentation framework to reason with inconsistent ontologies and Section 5 considers using argumentation framework to manage ontologies. Section 6 discusses two practical ontologies as experiments. Section 7 compares our scenario with existing proposals. In the last section, we summarize our work and discuss the future work.

2 Description logic $\mathcal{ALC}$

Description logic (DL) is a well-known family of knowledge representation formalisms. For more comprehensive background knowledge, we refer the reader to Chapter 2 of the DL Handbook (Baader et al. 2003).

DLs are fragments of FOL. That is, they can be translated into FOL. They are different from their predecessors such as semantic networks and frames since they are equipped with a formal, logic-based semantics. In DLs, elementary descriptions are *concept names* (unary predicates) and *role names* (binary predicates). Complex descriptions are built from them inductively using concept and role constructors provided by the particular DLs under consideration.

In this paper, we consider the DL $\mathcal{ALC}$ which is a simple yet relatively expressive DL, where $\mathcal{AL}$ is the abbreviation of attributive language and $\mathcal{C}$ denotes "complement". Let N_C and N_R be pairwise disjoint and countably infinite sets of *concept names* and *role names* respectively. Let N_I be an infinite set of *individual names*. A *signature* is a finite set $\Sigma = N_C \cup N_R \cup N_I$. We use the letters A and B for concept names, the letter R for role names, and the letters C and D for concepts. $\top$ and $\bot$ denote the *universal concept* and the *bottom concept* respectively. The set of $\mathcal{ALC}$ concepts is the smallest set such that:

- every concept name is a concept;
- if C and D are concepts, R is a role name, then the following expressions are also concepts: $\neg C$ (*full negation*), $C \sqcap D$ (*concept conjunction*), $C \sqcup D$ (*concept disjunction*), $\forall R.C$ (*value restriction on role names*) and $\exists R.C$ (*existential restriction on role names*).

For example, the concept description $Person \sqcap Female$ is an $\mathcal{ALC}$-concept describing those persons that are female. Suppose *hasChild* is a role name, the concept description $Person \sqcap \forall hasChild.Female$ expresses those persons whose children are all female. The concept $\forall hasChild.\bot \sqcap Person$ describes those persons who have no children.

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty domain $\Delta^{\mathcal{I}}$ and a mapping $\cdot^{\mathcal{I}}$, which maps every concept to a subset of $\Delta^{\mathcal{I}}$, every role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and for all concepts C, D, role R, satisfies the following equations:

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} &= \emptyset^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &= \left\{x \mid \exists y, (x, y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\right\} \\
(\forall R.C)^{\mathcal{I}} &= \left\{x \mid \forall y, (x, y) \in R^{\mathcal{I}} \text{ implies } y \in C^{\mathcal{I}}\right\}
\end{aligned}$$

A general concept inclusion axiom (GCI) or *a (terminological) axiom* is an inclusion statement of the forms $C \sqsubseteq D$, where C and D are two (possibly complex) concepts (concepts, for short). It is the statement about how concepts are related to each other. We use $C \equiv D$ as an abbreviation for the symmetrical pair of GCIs $C \sqsubseteq D$ and $D \sqsubseteq C$, called a *concept equivalence*. An interpretation $\mathcal{I}$ validates a GCI $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, and it satisfies a GCI $C \equiv D$ if and only if $C^{\mathcal{I}} = D^{\mathcal{I}}$. A finite set of GCIs is called a *TBox* or *terminology*. We can also formulate statements about individuals. A *concept* (*role*) assertion has the form $C(a)$ $(R(a, b))$, where C is a concept, R a role name, and a, b *individual names*. An *ABox* consists of a finite set of concept assertions and role assertions. In an ABox, one describes a specific state of affairs of an application domain in terms of concepts and roles.

To give a semantics to ABoxes, we need to extend interpretations to individual names. For each individual name a, $\cdot^{\mathcal{I}}$ maps it to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies a concept assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and it satisfies a role assertion $R(a,b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$. GCIs, concept assertions and role assertions are called *axioms*. An interpretation $\mathcal{I}$ is a *model* of a DL axiom if it satisfies this axiom.

An *ontology* $\mathcal{O}$ consists of a TBox and an ABox. The *signature* of an ontology $\mathcal{O}$, denoted by $Sig(\mathcal{O})$, is a set of all concept names, role names and individual names occurring in $\mathcal{O}$. A sub-ontology $\mathcal{O}'$ of ontology $\mathcal{O}$ is an ontology whose axioms and assertions are in $\mathcal{O}$, also denoted by $\mathcal{O}' \subseteq \mathcal{O}$ (In this paper, we treat an ontology as a set of axioms). And $\mathcal{O}'$ is a *proper sub-ontology* of $\mathcal{O}$ if $\mathcal{O}' \subseteq \mathcal{O}$ and $\mathcal{O} \not\subseteq \mathcal{O}'$.

An interpretation $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$ if it satisfies every axiom in $\mathcal{O}$. Let $Mod(\mathcal{O})$ be a set of all models of $\mathcal{O}$. An ontology $\mathcal{O}$ is *consistent* if there exists a model of $\mathcal{O}$, that is, $Mod(\mathcal{O}) \neq \emptyset$. An ABox $\mathcal{A}$ is *consistent* with respect to a TBox $\mathcal{T}$ if there exists a common model of $\mathcal{T}$ and $\mathcal{A}$. Given an ontology $\mathcal{O}$ and a DL axiom ϕ, we say $\mathcal{O}$ *entails* ϕ, denoted by $\mathcal{O} \models \phi$, if every model of $\mathcal{O}$ is a model of ϕ, that is, $Mod(\mathcal{O}) \subseteq Mod(\{\phi\})$. Furthermore, given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, we say $\mathcal{O}_1$ *entails* $\mathcal{O}_2$, denoted by $\mathcal{O}_1 \models \mathcal{O}_2$, if every model of $\mathcal{O}_1$ is a model of $\mathcal{O}_2$, that is, $Mod(\mathcal{O}_1) \subseteq Mod(\mathcal{O}_2)$. $\mathcal{O}_1$ is *equivalent* to $\mathcal{O}_2$ if $\mathcal{O}_1 \models \mathcal{O}_2$ and $\mathcal{O}_2 \models \mathcal{O}_1$, that is, $Mod(\mathcal{O}_1) = Mod(\mathcal{O}_2)$. A concept C is *satisfiable* with respect to a TBox $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$; and *unsatisfiable* otherwise. A TBox $\mathcal{T}$ is *incoherent* if there exists an unsatisfiable concept name in $\mathcal{T}$; and *coherent* otherwise. Consistency and coherency are two important properties of DL ontologies.

3 Argumentation framework for description logic ontologies

This section starts to introduce an argumentation framework in $\mathcal{ALC}$ based on BH's argumentation framework and binary argumentation. This adaption of BH's framework to DL is not trivial. For DL ontologies, there are two kinds of conflicts, namely, incoherency and inconsistency, while the inconsistency is the only one kind of conflicts occurring in PL ontologies or FOL ontologies. Moreover, DL syntax constructors ($\sqcup, \sqcap, \forall, \exists$) do not connect axioms but concepts or roles. That is, the syntactic structure of axioms need to be considered in the definition of argumentation framework. Another challenge of this adaption is constructing an argument tree, which is the core element of this framework, since the method of spanning argument trees in PL or FOL is not always feasible in DL.

3.1 Arguments in description logic $\mathcal{ALC}$

First, we present some basic definitions of arguments in $\mathcal{ALC}$ and discuss some primary properties in this subsection. Let $\mathcal{L}$ be a language of $\mathcal{ALC}$. We use $\alpha, \beta, \gamma, \ldots$ to denote axioms, $\mathcal{O}, \Phi, \Psi, \ldots$ to denote sets of axioms. $\mathcal{O}$ is *finite* if the number of axioms in $\mathcal{O}$ is finite. In this paper, we mainly consider finite ontologies.

Following from argumentation in propositional logic (Besnard and Hunter 2001), an *argument* for an axiom ϕ is a pair $\langle \Phi, \phi \rangle$ satisfying the following three conditions:

1. Φ is a sub-ontology of $\mathcal{O}$;

2. Φ is consistent and coherent;
3. $\Phi \models \phi$; and there is no Φ' which is a sub-ontology of Φ such that $\Phi' \models \phi$.

If $\mathbf{A} = \langle \Phi, \phi \rangle$ is an *argument*, we say that $\mathbf{A}$ is an argument for ϕ in $\mathcal{O}$ and we also say that Φ supports ϕ. Without causing any confusion, we briefly say arguments. In addition, we call Φ the *support* of $\mathbf{A}$ and ϕ the *consequent* of $\mathbf{A}$. $Sup(\mathbf{A}) = \Phi$ and $Con(\mathbf{A}) = \phi$.

Intuitively, the support of an argument for an axiom is a minimal *prime implicate* (see Bienvenu 2008) of it which is consistent and coherent.

To capture arguments for conjunction of DL axioms, we need an extended notion of "arguments for a set of axioms" since there exists not always a DL axiom to express a set of DL axioms, contrary to PL formulas.

Definition 1 Let Ψ be a set of axioms. An *argument* for Ψ is a pair $\langle \Phi, \Psi \rangle$ such that Φ is a consistent and coherent sub-ontology of $\mathcal{O}$ which $\Phi \models \Psi$ and there is no Φ' which is a sub-ontology of Φ such that $\Phi' \models \Psi$. Let $Arg(\mathcal{O})$ be the set of all arguments whose support are in $\mathcal{O}$.

The support and consequent are analogously defined. For instance, $\langle \Phi, \Psi \rangle$ where $\Phi = \{R(a, b)\}$ and $\Psi = \{\exists R.\top(a), \exists R^{-}.\top(b)\}$ is an argument for Ψ.

The notion of arguments for an axiom is a special case of the notion of arguments for a set of axioms. In this sense, we also use $\langle \Phi, \{\phi\} \rangle$ to denote $\langle \Phi, \phi \rangle$.

Given two arguments $\mathbf{A}'$ and $\mathbf{A}''$, $\mathbf{A}'$ is *equal to* $\mathbf{A}''$, denoted by $\mathbf{A}' = \mathbf{A}''$, if $Sup(\mathbf{A}') = Sup(\mathbf{A}'')$ and $Con(\mathbf{A}') = Con(\mathbf{A}'')$.

For any non-empty ontology $\mathcal{O}$, we say $\langle \{\phi\}, \{\phi\} \rangle$ for any $\phi \in \mathcal{O}$ is a *basic argument*. A *trivial argument* is in form of $\langle \emptyset, \Upsilon \rangle$ where Υ is a set of some tautologies.

Example 1 Let $\mathcal{O}_1 = (\mathcal{T}_1, \mathcal{A}_1)$ be an ontology where $\mathcal{T}_1 = \{Penguin \sqsubseteq Bird, Bird \sqsubseteq Fly, Penguin \sqsubseteq \neg Fly, DegradedBird \sqsubseteq \neg Fly, Swallow \sqsubseteq Bird, Penguin \sqsubseteq \exists hasFood.Fish, Swallow \sqsubseteq \neg Penguin\}$ and $\mathcal{A}_1 = \{Penguin(tweety), DegradedBird(tweety), Swallow(slikken)\}$. From $\mathcal{O}_1$, Ontology $\mathcal{O}$ tells us as follows: penguins are birds; birds can fly; penguins cannot fly; degraded birds can not fly; swallows are birds; penguins eat some fish; swallows are not penguins; *tweety* is a penguin; *tweety* is a degraded bird; and *slikken* is a swallow.

Some of the arguments are as follows: let *t* be the acronym of "*tweety*" and *s* the acronym of "*slikken*",

$$\mathbf{A}_1 = \langle \{Penguin(t)\}, \{Penguin(t)\} \rangle$$

$$\mathbf{A}_2 = \langle \{Swallow(s)\}, \{Swallow(s)\} \rangle$$

$$\mathbf{A}_3 = \langle \{Penguin(t), Penguin \sqsubseteq Bird\}, \{Bird(t)\} \rangle$$

$$\mathbf{A}_4 = \langle \{Swallow(s), Swallow \sqsubseteq Bird\}, \{Bird(s)\} \rangle$$

$$\mathbf{A}_5 = \langle \{Penguin(t), Penguin \sqsubseteq \neg Fly\}, \{Penguin(t), \neg Fly(t)\} \rangle$$

$$\mathbf{A}_6 = \langle \{DegradedBird(t), DegradedBird \sqsubseteq \neg Fly\}, \{\neg Fly(t)\} \rangle$$

$$\mathbf{A}_7 = \langle \{Penguin(t), Penguin \sqsubseteq Bird, Bird \sqsubseteq Fly\}, \{Fly(t)\} \rangle$$

$$\mathbf{A}_8 = \langle \{Penguin(t), Penguin \sqsubseteq Bird, Bird \sqsubseteq Fly\}, \{Penguin(t), Fly(t)\} \rangle$$

$$\mathbf{A}_9 = \langle\{Penguin \sqsubseteq Bird\}, \{Penguin \sqsubseteq Bird\}\rangle$$

$$\mathbf{A}_{10} = \langle\{Bird \sqsubseteq Fly\}, \{Bird \sqsubseteq Fly\}\rangle$$

$$\mathbf{A}_{11} = \langle\{Penguin \sqsubseteq \neg Fly\}, \{Penguin \sqsubseteq \neg Fly\}\rangle$$

$$\mathbf{A}_{12} = \langle\{Penguin \sqsubseteq Bird, Bird \sqsubseteq Fly\}, \{Penguin \sqsubseteq Fly\}\rangle$$

$$\mathbf{A}_{13} = \langle\{Penguin(t), Penguin \sqsubseteq \exists hasFood.Fish\}, \{\exists hasFood.Fish(t)\}\rangle$$

$$\mathbf{A}_{14} = \langle\{Penguin(t), Swallow(s), Swallow \sqsubseteq \neg Penguin\}, \{\neg Swallow(t), \neg Penguin(s)\}\rangle$$

In Example 1, $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_9$, $\mathbf{A}_{10}$ and $\mathbf{A}_{11}$ are basic arguments. Note that each arguments $\mathbf{A}_i$ except for $\mathbf{A}_5$, $\mathbf{A}_8$ and $\mathbf{A}_{14}$ is indeed an argument for an axiom. The consequent of $\mathbf{A}_{14}$ can not be represented by a DL axiom.

Arguments often interrelate with each other. In Example 1, the consequents of both $\mathbf{A}_5$ and $\mathbf{A}_6$ are identical and the supports of both $\mathbf{A}_7$ and $\mathbf{A}_8$ are identical. Besides, the union of consequents of both $\mathbf{A}_5$ and $\mathbf{A}_7$ is inconsistent and the union of both the support of $\mathbf{A}_{11}$ and the consequent of $\mathbf{A}_{12}$.

Next, we introduce another relationship to capture a relation that arguments "attack" each other (Dung 1995b).

Definition 2 Let $\mathcal{O}$ be an ontology. Given two arguments in $\mathcal{O}$ $\mathbf{A} = \langle\Phi, \Psi\rangle$ and $\mathbf{A}' = \langle\Phi', \Psi'\rangle$, $\mathbf{A}'$ is an *undercut* of $\mathbf{A}$ if and only if $\Psi' \cup \Phi$ is inconsistent or incoherent. Let $Undercuts(\mathcal{O}, \mathbf{A})$ be the set of all undercuts of $\mathbf{A}$ in $\mathcal{O}$.

Intuitively, the undercut of an argument is against the argument. In this sense, the consequent of its undercut *invalidates* the support of it, that is, the union of the consequent and the support is inconsistent or incoherent.

In Example 1, $\mathbf{A}_7$, $\mathbf{A}_8$ are undercuts of both $\mathbf{A}_5$ and $\mathbf{A}_6$ which are also undercuts of $\mathbf{A}_7$, $\mathbf{A}_8$. $\mathbf{A}_{11}$ is an undercut of $\mathbf{A}_{12}$ and $\mathbf{A}_{12}$ is an undercut of $\mathbf{A}_{12}$.

However, it is not always true that for any arguments $\mathbf{A}$ and $\mathbf{A}'$, if $\mathbf{A}'$ is an undercut of $\mathbf{A}$ then $\mathbf{A}$ is an undercut of $\mathbf{A}'$. For instance, two arguments $\mathbf{A} = \langle\{A(a), A \sqsubseteq B\}, \{B(a)\}\rangle$ and $\mathbf{A}' = \langle\{C(a), C \sqsubseteq \neg A\}, \{\neg A(a)\}\rangle$. We find that $\mathbf{A}'$ is an undercut of $\mathbf{A}$ while $\mathbf{A}$ is not an undercut of $\mathbf{A}'$

Next result shows that if an ontology is inconsistent or incoherent, then there exists some undercuts of some arguments.

Proposition 1 *Let $\mathcal{O}$ be an ontology and $\mathbf{A} = \langle\Phi', \Psi'\rangle$ an argument. If Φ'' is a consistent and coherent sub-ontology of $\mathcal{O}$ such that $\Phi' \cup \Phi''$ is inconsistent or incoherent, then $Undercuts(\mathbf{A}) \neq \emptyset$, that is, there exists an undercut $\mathbf{A}_u$ of $\mathbf{A}$.*

Proof To prove it, we only need to construct such an undercut $\mathbf{A}_u$. If $\Phi' \cup \Phi''$ is inconsistent or incoherent then there exists a set of axioms Ψ'' such that $\Phi'' \models \Psi''$ and $\Phi' \cup \Psi''$ is inconsistent or incoherent. Then there exists a minimal sub-ontology $\Phi^{(3)} \subseteq \Phi''$ such that $\Phi^{(3)} \models \Psi''$. Let $\mathbf{A}_u = \langle\Phi^{(3)}, \Psi''\rangle$. Thus $\mathbf{A}_u$ is an undercut of $\mathbf{A}$. □

Proposition 1 also provides a way to compute undercuts of a given argument.

Undercut is a basic notion in defining BH's framework (Besnard and Hunter 2001). This framework is built on argument trees where arguments are taken as nodes and each edge $\langle \mathbf{A}_u, \mathbf{A} \rangle$ means that $\mathbf{A}_u$ "attacks" $\mathbf{A}$ (i.e., $\mathbf{A}_u$ is an undercut of $\mathbf{A}$ here).

A challenge of adapting BH's framework for $\mathcal{ALC}$ ontologies is that argument trees based on undercuts become infinite since the number of models of some $\mathcal{ALC}$ ontologies is infinite. For instance, let $\mathcal{O} = (\{A \sqsubseteq \exists R.A\}, \{A(a)\})$, we find that the number of models of $\mathcal{O}$ is infinite (see Baader et al. 2003; Calvanese 1996).

To address this challenge, it is necessary to investigate how to select finite number of undercuts to represent all undercuts of an argument.

Firstly, we show that undercuts of an argument mainly depend on the support of the argument what they "attack".

Proposition 2 *Let $\mathcal{O}$ be an ontology and $\mathbf{A}', \mathbf{A}''$ two arguments. If $Sup(\mathbf{A}') \subseteq Sup(\mathbf{A}'')$ then $Undercuts(\mathcal{O}, \mathbf{A}') \subseteq Undercuts(\mathcal{O}, \mathbf{A}'')$, that is, for any argument $\mathbf{A}_u$, if $\mathbf{A}_u$ is an undercut of $\mathbf{A}'$ then $\mathbf{A}$ is an undercut of $\mathbf{A}''$.*

Proof Because $Sup(\mathbf{A}') \subseteq Sup(\mathbf{A}'')$, for any argument $\mathbf{A}_u$, if $\mathbf{A}_u$ is an undercut of $\mathbf{A}'$ then $Con(\mathbf{A}_u) \cup Sup(\mathbf{A}')$ is inconsistent or incoherent by Definition 2. Because $Sup(\mathbf{A}') \subseteq Sup(\mathbf{A}'')$, $Con(\mathbf{A}_u) \cup Sup(\mathbf{A}'')$ is inconsistent or incoherent. Then $\mathbf{A}_u$ is an undercut of $\mathbf{A}''$ by Definition 2. Therefore, $Undercuts(\mathcal{O}, \mathbf{A}') \subseteq Undercuts(\mathcal{O}, \mathbf{A}'')$. □

We then consider how an argument encompasses the other by using the inclusion relation between supports.

Definition 3 Let $\mathbf{A}, \mathbf{A}'$ be two arguments. We say $\mathbf{A}'$ *encompasses* $\mathbf{A}$, denoted by $\mathbf{A} \preceq_e \mathbf{A}'$, if $Sup(\mathbf{A}) \subseteq Sup(\mathbf{A}')$. $\mathbf{A}'$ *strictly encompasses* $\mathbf{A}$, denoted by $\mathbf{A} \prec_e \mathbf{A}'$ if $Sup(\mathbf{A}) \subseteq Sup(\mathbf{A}')$ but $Sup(\mathbf{A}') \not\subseteq Sup(\mathbf{A})$.

Note that the encompass relation is a partially preferential relation between two arguments. The encompass relation $\preceq_e$ ($\prec_e$) is transitive. That is, $\mathbf{A} \preceq_e \mathbf{A}'$ and $\mathbf{A}' \preceq_e \mathbf{A}''$ implies $\mathbf{A} \preceq_e \mathbf{A}''$.

In Example 1, $\mathbf{A}_1 \prec_e \mathbf{A}_3$, $\mathbf{A}_3 \prec_e \mathbf{A}_7$, $\mathbf{A}_9 \prec_e \mathbf{A}_3$, $\mathbf{A}_{11} \prec_e \mathbf{A}_5$, $\mathbf{A}_2 \prec_e \mathbf{A}_{14}$.

In fact, all basic arguments only encompass trivial arguments. Let $\mathbf{A}$ be a basic argument. If there exists an argument $\mathbf{A}'$ such that $\mathbf{A}' \prec_e \mathbf{A}$ then $\mathbf{A}'$ is a trivial argument, that is, $Sup(\mathbf{A}') = \emptyset$.

Given an argument, we can introduce the encompass relation over its undercuts since each undercut is an argument. We say those undercuts which are not encompassed by any other undercuts *minimally encompass undercuts* formally defined as follows:

Definition 4 Let $\mathcal{O}$ be an ontology and $\mathbf{A}$ an argument. A *minimally encompass undercut* (MEU, for short) $\mathbf{A}_u$ of $\mathbf{A}$ is an undercut of $\mathbf{A}$ and there exists no undercut $\mathbf{A}'_u$ of $\mathbf{A}$ such that $\mathbf{A}_u \prec_e \mathbf{A}'_u$. We use $MEU(\mathcal{O}, \mathbf{A})$ to denote the set of all MEUs of $\mathbf{A}$ in $\mathcal{O}$.

Intuitively, MEUs are undercuts whose supports are minimal sub-ontologies (i.e., they contain minimal number of axioms). In Example 1, $\mathbf{A}_5$ and $\mathbf{A}_6$ are MEUs of both $\mathbf{A}_7$ and $\mathbf{A}_8$. $\mathbf{A}_7$ and $\mathbf{A}_8$ are MEUs of both $\mathbf{A}_5$ and $\mathbf{A}_6$. $\mathbf{A}_{11}$ is a MEU of $\mathbf{A}_{12}$ and $\mathbf{A}_{12}$ is a MEU of $\mathbf{A}_{11}$.

MEUs with different supports are incomparable with each other.

Proposition 3 *Let $\mathcal{O}$ be an ontology and $\mathbf{A}$ an argument. For any two arguments $\mathbf{A}'_u, \mathbf{A}''_u \in MEU(\mathcal{O}, \mathbf{A})$, if $Sup(\mathbf{A}'_u) \neq Sup(\mathbf{A}''_u)$ then $\mathbf{A}'_u \not\preceq_e \mathbf{A}''_u$ and $\mathbf{A}''_u \not\preceq_e \mathbf{A}'_u$.*

Proof We have that neither $Sup(\mathbf{A}'_u) \subseteq Sup(\mathbf{A}''_u)$ nor $Sup(\mathbf{A}''_u) \subseteq Sup(\mathbf{A}'_u)$ by Definition 4 since $Sup(\mathbf{A}'_u) \neq Sup(\mathbf{A}''_u)$. Thus $\mathbf{A}'_u \not\preceq_e \mathbf{A}''_u$ and $\mathbf{A}''_u \not\preceq_e \mathbf{A}'_u$ by Definition 3. □

In Example 1, $\mathbf{A}_5$, $\mathbf{A}_6$ are incomparable with each other.

There exist multiple different MEUs with the same support since multiple consequents can be brought by a support. In Example 1, $\mathbf{A}_7$ and $\mathbf{A}_8$ are two different MEUs of $\mathbf{A}_5$ while $Sup(\mathbf{A}_7) = Sup(\mathbf{A}_8)$.

Proposition 2 still holds by considering MEUs instead of undercuts.

Proposition 4 *Let $\mathcal{O}$ be an ontology and $\mathbf{A}', \mathbf{A}''$ two arguments. If $Sup(\mathbf{A}') \subseteq Sup(\mathbf{A}'')$ then $MEU(\mathcal{O}, \mathbf{A}') \subseteq MEU(\mathcal{O}, \mathbf{A}'')$, that is, for any argument $\mathbf{A}_u$, $\mathbf{A}_u$ is a MEU of $\mathbf{A}'$ if and only if $\mathbf{A}$ is a MEU of $\mathbf{A}''$.*

Proof Because $Sup(\mathbf{A}') = Sup(\mathbf{A}'')$, $Undercuts(\mathcal{O}, \mathbf{A}') \subseteq Undercuts(\mathcal{O}, \mathbf{A}'')$ by Proposition 4. Then $MEU(\mathcal{O}, \mathbf{A}') \subseteq MEU(\mathcal{O}, \mathbf{A}'')$ by Definition 4. □

Two arguments with the same support have the same MEUs. In this sense, supports can be used to partition all MEUs of some argument into some classes.

Formally, let $\mathbf{O}$ be an ontology and $\mathbf{A}$ an argument in $\mathcal{O}$, given a sub-ontology Φ, we denote

$$Arg^u(\mathcal{O}, \mathbf{A}, \Phi) = \{\mathbf{A}_u \mid Sup(\mathbf{A}_u) = \Phi \text{ and } \mathbf{A}_u \in MEU(\mathcal{O}, \mathbf{A})\}$$

$$\Omega^u(\mathcal{O}, \mathbf{A}) = \{\Phi \in \mathcal{O} \mid Arg^u(\mathcal{O}, \mathbf{A}, \Phi) \neq \emptyset\}$$

We say $Arg^u(\mathcal{O}, \mathbf{A}, \Phi)$ is a *class* of $MEU(\mathcal{O}, \mathbf{A})$ w.r.t. Φ and $\Omega^u(\mathcal{O}, \mathbf{A})$ is a *partition* of $MEU(\mathcal{O}, \mathbf{A})$.

In Example 1, $\mathbf{A}_7$ and $\mathbf{A}_8$ belong to the class $Arg^u(\mathcal{O}_1, \mathbf{A}_5, \Phi)$ where $\Phi = Sup(\mathbf{A}_7)$.

Given a finite ontology $\mathcal{O}$, though $\mid Arg^u(\mathcal{O}, \mathbf{A}, \Phi) \mid$ is possibly infinite while $\mid Arg(\mathcal{O}, \Phi) \mid$ is always finite since $\Omega^u(\mathcal{O}, \mathbf{A}) \subseteq 2^{\mathcal{O}}$ where $2^{\mathcal{O}}$ is the power set of $\mathcal{O}$, that is, $2^{\mathcal{O}} = \{\mathcal{O}' \mid \mathcal{O}' \subseteq \mathcal{O}\}$.

Though different MEUs (of an argument) in a class have different consequents, all of their consequents invalidate the support. Next, we introduce a *canonical undercut* to characterize such a common feature of all MEUs in a class.

Definition 5 Let $\mathcal{O}$ be an ontology and $\mathbf{A}$ be an argument. Given a sub-ontology Φ, we say $\langle \Phi, \diamond \rangle$ is a *canonical undercut* if $Arg^u(\mathcal{O}, \mathbf{A}, \Phi) \neq \emptyset$.

Here we use $\diamond$ to express the existence of consequent of such a MEU. We use $UC(\mathcal{O}, \mathbf{A})$ to denote the set of all canonical undercuts of $\mathbf{A}$ in $\mathcal{O}$.

Intuitively, a canonical undercut represents all MEUs whose supports are identical. Note that different canonical undercuts have different supports.

In Example 1, we use $\mathbf{A}_i^*$ $(1 \leq i \leq 14)$ to denote an argument $\langle \Phi_i, \diamond \rangle$ where $\Phi_i = Sup(\mathbf{A}_i)$. We find that $\mathbf{A}_7^* = \mathbf{A}_8^*$ and $\mathbf{A}_7^*$ is a canonical undercut of both $\mathbf{A}_5$ and $\mathbf{A}_6$. Both $\mathbf{A}_5^*$ and $\mathbf{A}_6^*$ are canonical undercuts of $\mathbf{A}_7$. $\mathbf{A}_{11}^*$ is a canonical undercut of $\mathbf{A}_{12}$ and $\mathbf{A}_{12}^*$ is a canonical undercut of $\mathbf{A}_{11}$.

Proposition 5 *Let $\mathcal{O}$ be an ontology. For any argument $\mathbf{A}$, the number of canonical undercuts of $\mathbf{A}$ is finite.*

Proof Because all canonical undercuts are in form of $\langle \Phi, \diamond \rangle$ where Φ is a sub-ontology of $\mathcal{O}$. The number of sub-ontologies of a finite ontology $\mathcal{O}$ is finite. Therefore, the number of canonical undercuts of $\mathbf{A}$ is finite. □

To take advantage of finiteness of canonical undercuts, we develop a naive algorithm to constructing canonical undercuts shown in Algorithm 1 where Λ stores all sub-ontologies which invalidate the support of $\mathbf{A}$ and Π stores all canonical undercuts of $\mathbf{A}$.

Algorithm 1 Naive algorithm for constructing canonical undercuts

```
1:  procedure NACCU$\mathcal{O}$, $\mathbf{A}$
2:      $2^{\mathcal{O}}$: the power set of $\mathcal{O}$
3:      $\Lambda = \{\}$
4:      $\Pi = \{\}$
5:      for each sub-ontology $\Phi \in 2^{\mathcal{O}}$
6:          it $\Phi \cup Sup(\mathbf{A})$ is inconsistent or incoherent
7:              $\Lambda = \Lambda \cup \{\Phi\}$
8:          end if
9:      end for
10:     for each sub-ontology $\Phi \in \Lambda$
11:         if there exists no $\Psi \in \Lambda$ such that $\Psi \subset \Phi$
12:             $\Pi = \Pi \cup \{\langle \Phi, \diamond \rangle\}$
13:         end if
14:     end for
15:     return $\Pi$
16: end procedure
```

Next result shows that Algorithm 1 is sound and complete.

Proposition 6 *Let $\mathcal{O}$ be an ontology. For any argument $\mathbf{A}$, we have $CU(\mathcal{O}, \mathbf{A}) = NACCU(\mathcal{O}, \mathbf{A})$.*

Proof Firstly, we show that Algorithm 1 is complete. For any $\mathbf{A}_u \in CU(\mathcal{O}, \mathbf{A})$, then $Sup(\mathbf{A}_u) \cup Sup(\mathbf{A})$ is inconsistent or incoherent by Definition 2. Thus $Sup(\mathbf{A}_u) \in \Lambda$ in Algorithm 1. Because $\mathbf{A}_u$ is the support of some MEU of $\mathbf{A}$, there exists no other Ψ such that $\Psi \subset Sup(\mathbf{A}_u)$ and $\langle \Psi, \diamond \rangle \in CU(\mathcal{O}, \mathbf{A})$ by Definition 5. That is, there exists no other $\Psi \in \Lambda$ such that $\Psi \subset Sup(\mathbf{A}_u)$ by Definition 4. Thus $Sup(\mathbf{A}_u) \in \Pi$.

Secondly, we show that Algorithm 1 is sound. For any $\langle \Phi, \diamond \rangle \in \text{NACCU}(\mathcal{O}, \mathbf{A})$, we need to show that $\langle \Phi, \diamond \rangle \in CU(\mathcal{O}, \mathbf{A})$. Because $\langle \Phi, \diamond \rangle \in \text{NACCU}(\mathcal{O}, \mathbf{A})$ we have $\Phi \in \Pi$ and $\Phi \in \Lambda$ in Algorithm 1. $\Phi \in \Lambda$ means that $\Phi \cup Sup(\mathbf{A})$ is inconsistent or incoherent. Thus there exists some undercut $\mathbf{A}_u$ of $\mathbf{A}$ such that $Sup(\mathbf{A}_u) = \Phi$ by

Proposition 1. Assume that $\mathbf{A}_u$ is not a MEU of $\mathbf{A}$. There exists a MEU $\mathbf{A}'_u$ of $\mathbf{A}$ such that $Sup(\mathbf{A}'_u) \subset \Phi$. Because $\mathbf{A}'_u$ is also an undercut of $\mathbf{A}$, $Sup(\mathbf{A}'_u) \cup Sup(\mathbf{A})$ is inconsistent or incoherent. That is, $Sup(\mathbf{A}'_u) \in \Lambda$. Thus we find a sub-ontology $Sup(\mathbf{A}'_u) \in \Lambda$ such that $Sup(\mathbf{A}'_u) \subset \Phi$ which contradicts $\Phi \in \Pi$. □

3.2 Argumentation framework in description logic $\mathcal{ALC}$

In this subsection, we define *argument trees* by using canonical undercuts and then introduce BH's argumentation framework for DL ontologies.

An important target of Besnard and Hunter's framework is providing a dialogue mechanism in the shape of trees, called *argument trees*. An argument tree is a tree, whose nodes are arguments and edges represent the "attack" relation between arguments / nodes, formally defined as follows:

Definition 6 Let $\mathcal{O}$ be an ontology and $\mathbf{A}$ an argument in $\mathcal{O}$. An *argument tree* of $\mathbf{A}$ w.r.t. $\mathcal{O}$ is a tree where the nodes are arguments such that

1. the root is $\mathbf{A}$;
2. for no node $\mathbf{A}'$ with ancestor nodes $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(n)}$, is $Sup(\mathbf{A}')$ a sub-ontology of $Sup(\mathbf{A}^{(1)}) \cup \ldots \cup Sup(\mathbf{A}^{(n)})$;
3. the children nodes of a node $\mathbf{A}$ consist of all canonical undercuts of $\mathbf{A}$, which obeys Item 2.

Intuitively, Item 1 shows that an argument tree for an axiom starts an argument for this axiom; Item 3 shows that the argument tree grows by adding canonical undercuts of the current nodes as its children nodes recursively till termination; and Item 2 states that each newly added children should contain some new axioms not occurring in its ancestors. In other words, when $\mathcal{O} \subseteq Sup(\mathbf{A}_{root}) \cup Sup(\mathbf{A}^{(1)}) \cup \ldots \cup Sup(\mathbf{A}^{(n)})$, argument trees will terminate its growing since $\mathcal{O}$ is finite. In this sense, Item 2 ensures the finite depth of argument trees.

As a direct result, argument trees are finite if it has a finite number of branches and a finite depth.

Proposition 7 *Any argument tree of an arbitrary argument w.r.t. a finite ontology is finite.*

Proof Let $\mathcal{O}$ be an ontology and $\mathbf{A}$ an argument in $\mathcal{O}$. Since $\mathcal{O}$ is finite, the number of sub-ontologies of $\mathcal{O}$ is finite. In an argument tree, no branch can be infinite by Item 2 of Definition 6 (discussed above). Also, the number of canonical undercuts is finite by Proposition 5. The branching factor in an argument tree is finite by Item 3 of Definition 6. □

In particular, all argument trees w.r.t. consistent and coherent ontologies have a distinguished feature.

Proposition 8 *If an ontology $\mathcal{O}$ is consistent and coherent, then all argument trees w.r.t. $\mathcal{O}$ have exactly one node.*

Proof Let $\mathbf{A}$ be a random argument in $\mathcal{O}$. Assume there is an argument tree T of $\mathbf{A}$ w.r.t. $\mathcal{O}$ which has two nodes in $\mathcal{O}$ $\Rightarrow$ the child $\mathbf{A}'$ of $\mathbf{A}$ (i.e., the root of T) is a canonical undercut for $\mathbf{A}$ $\Rightarrow$ $Sup(\mathbf{A}') \cup Sup(\mathbf{A}) \subseteq \mathcal{O}$ is inconsistent or incoherent which contradicts the fact that $\mathcal{O}$ is consistent and coherent. □

Proposition 8 can be used to diagnose whether an ontology is consistent and coherent.

To characterize ontology consistency and coherency, two axiom negations called *consistency-negation* and *coherency-negation* are introduced in Flouris et al. (2006) where their existence and decidability are proved.

Let ϕ be an axiom. An axiom is called the *consistency-negation* of ϕ, denoted by $\neg\phi$ if and only if

1. $\{\phi, \neg\phi\}$ is inconsistent;
2. There exist no other ψ such that ψ satisfies Item 1 and $\{\psi\} \models \{\phi\}$ but $\{\phi\} \not\models \{\psi\}$.

An axiom is called the *coherency-negation* of ϕ, denoted by $-\phi$ if and only if

1. $\{\phi, \neg\phi\}$ is incoherent;
2. There exist no other ψ such that ψ satisfies Item 1 and $\{\psi\} \models \{\phi\}$ but $\{\phi\} \not\models \{\psi\}$.

For instance (see Flouris et al. 2006), let us consider the consistency negation and the coherence negation of an axiom $C \sqsubseteq D$, where C and D are named concepts.

$$\neg(C \sqsubseteq D) = \exists(C \sqcap \neg D), \quad -(C \sqsubseteq D) = C \sqsubseteq \neg D$$

where $\exists(C \sqcap \neg D)$ is an existence axiom (see Horrocks and Patel-Schneider 2003), which states there exists some instance of the concept $C \sqcap \neg D$, that is, for any individual a, $\{C(a), \neg D(a)\} \models \exists(C \sqcap \neg D)$. Technically, we can treat existence axioms as queries over ontologies. Note that, in any ontologies containing $C \sqsubseteq D$ and $C \sqsubseteq \neg D$, the concept C is unsatisfiable.

Note that undercuts are related to either ontology inconsistency or ontology incoherency. For convenience, we directly use $\sim \phi$ to denote either $\neg\phi$ or $-\phi$. Note that $\sim A(a)$ is $\neg A(a)$ for any concept name A and any individual name a.

In Example 1, $\mathbf{A}_5$ is an argument for $\neg(Penguin \sqsubseteq Fly)$ and $\mathbf{A}_{12}$ is an argument for $-(Penguin \sqsubseteq Fly)$. Thus both $\mathbf{A}_5$ and $\mathbf{A}_{12}$ are arguments for $\sim (Penguin \sqsubseteq Fly)$.

Now we are ready to introduce our argumentation framework of some axioms in a given ontology.

Definition 7 Let $\mathcal{O}$ be an ontology and ϕ an axiom. An *argumentation framework* of ϕ w.r.t. $\mathcal{O}$, defined as $AF(\mathcal{O}, \phi)$, is a pair

$$\langle ArgTree(\mathcal{O}, \phi), ArgTree(\mathcal{O}, \sim \phi)\rangle \tag{1}$$

where $ArgTree(\mathcal{O}, \psi) = \{T \mid T$ is an argument tree of $\mathbf{A}$ w.r.t. $\mathcal{O}$ and $\mathbf{A}$ is an argument for ψ w.r.t. $\mathcal{O}\}$.

We also say *binary argumentation framework* since it is a pair here.

An argument tree is *for* (*against*) an axiom ϕ if its root is an argument for ϕ ($\sim \phi$).

Intuitively, $ArgTree(\mathcal{O}, \phi)$ is a set of all argument trees for ϕ and $ArgTree(\mathcal{O}, \sim \phi)$ is a set of all argument trees against ϕ.

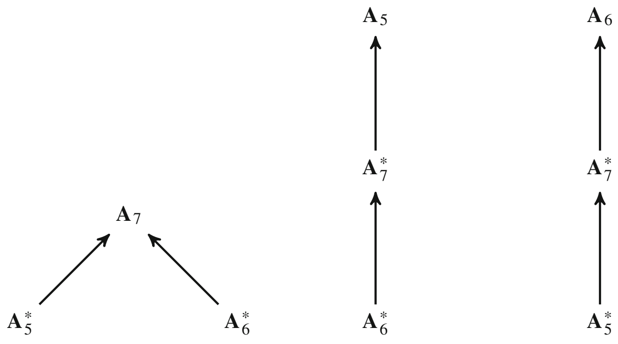

Fig. 1 Argument trees: T_1 (*left*), T_2 (*middle*), T_3 (*right*)

In Example 1, $AF(\mathcal{O}_1, Fly(tweety)) = \langle\{T_1\}, \{T_2, T_3\}\rangle$ where T_1 is an argument tree for $Fly(tweety)$ and T_2, T_3 are argument trees against $Fly(tweety)$ (shown in Fig. 1). $AF(\mathcal{O}_1, Penguin \sqsubseteq Fly) = \langle\{T_4\}, \{T_5, T_6\}\rangle$ where T_4 is an argument tree for $Penguin \sqsubseteq Fly$ and T_5, T_6 are argument trees against $Penguin \sqsubseteq Fly$ (shown in Fig. 2).

Proposition 9 *Let $\mathcal{O}$ be a consistent and coherent ontology and ϕ an axiom. If $\mathcal{O} \models \phi$ then*

1. *for each tree $T \in ArgTree(\mathcal{O}, \phi)$, T has exactly one node (root);*
2. *$ArgTree(\mathcal{O}, \sim \phi) = \emptyset$.*

Proof Because $\mathcal{O}$ is consistent and coherent, we have for any argument tree T in $ArgTree(\mathcal{O}, \phi)$, T has exactly one node by Proposition 8. Next, we only show that $ArgTree(\mathcal{O}, \sim \phi)$ of ϕ is the empty set. Suppose that $ArgTree(\mathcal{O}, \sim \phi) \neq \emptyset$, i.e., there exists an argument tree T for $\sim \phi$. Let $\mathbf{A} = \langle\Psi, \sim \phi\rangle$ be the root of T and $\mathbf{A}' = \langle\Phi, \phi\rangle$ the root of an argument tree T' in $ArgTree(\mathcal{O}, \phi)$. Then $\Psi \cup \Phi$ is inconsistent or incoherent. So there exists a canonical undercut $\mathbf{A}_u$ for ϕ by Definition 5. Thus T' contains at least a child node $\mathbf{A}'$ except for the root node, that is, there exists an argument tree for ϕ that contains at least two nodes. It contradicts the precondition. □

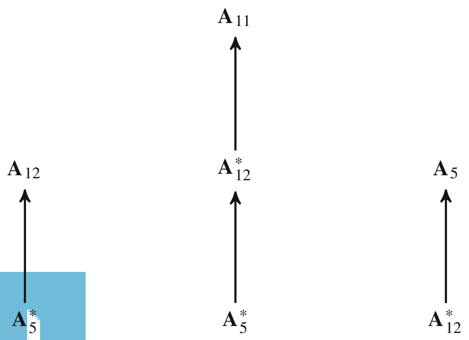

Fig. 2 Argument trees: T_4 (*left*), T_5 (*middle*), T_6 (*right*)

4 Reasoning with inconsistent ontologies by using argumentation

A basic task in ontology reasoning is to decide whether an axiom is accepted by its argumentation framework in a given ontology. As discussed earlier, it might be trivial if ontologies are inconsistent or incoherent. In this section, based on our argumentation framework, we propose a non-classical semantics to develop an inconsistency-tolerant reasoning with ontologies with conflicts and obtain meaningful conclusions.

Compared with using models to characterize the classical semantics, we use the (semantic) entailment to characterize our non-classical semantics. To do this, we first define the acceptance of axioms in our semantic entailment.

Let $\mathbf{A}$, $\mathbf{A}'$ and $\mathbf{A}''$ be three arguments. If $\mathbf{A}$ is undercut by $\mathbf{A}'$ and $\mathbf{A}'$ is undercut by $\mathbf{A}''$ then $\mathbf{A}''$ is called a *defence* for $\mathbf{A}$. We define the "*defend*" relation as the transitive closure of "*being a defence*". An argument tree is said to be *successful* if and only if every leaf defends the root node. An argument is called *self-protected* if there exists a successful argument tree for it. An axiom is *accepted* if and only if there exists a self-protected argument for it.

In Example 1, T_2, T_3, T_5 are successful and T_1, T_4 are not successful. $\mathbf{A}_5$, $\mathbf{A}_{11}$ are self-protected while $\mathbf{A}_7$, $\mathbf{A}_6$ are not self-protected. $Fly(tweety)$ is accepted and $Pegnuin \sqsubseteq Fly$ is not accepted.

Next, based on the argumentation framework, we define an inference relation between ontologies and axioms and, for easy understanding, we still say it an entailment (relation) which is normally characterizing the inclusion relation between models of two ontologies.

Definition 8 Let $\mathcal{O}$ be an ontology and ϕ an axiom. We say $\mathcal{O}$ *argumentatively entails* (a-entails, for short) ϕ, denoted by $\mathcal{O} \models_a \phi$, if there exists a successful argument tree for ϕ. In this case, we call $\models_a$ *argumentative entailment (relationship)* (a-entailment, for short).

Two basic inference problems, namely, *instance checking* and *subsumption checking* in ontology reasoning under our semantics are correspondingly defined as follows:

1. *instance checking*: given a concept C and an individual a, a is an *argumentative instance* (*a-instance*, for short) of concept if $\mathcal{O} \models_a C(a)$.
2. *subsumption checking*: a concept D *argumentatively subsumes* (*a-subsumes*, for short) a concept C if $\mathcal{O} \models_a C \sqsubseteq D$.

In Example 1, $\mathcal{O}_1 \models_a \neg Fly(tweety)$ and $\mathcal{O}_1 \models_a Penguin \sqsubseteq \neg Fly$.

In the following, we enumerate several good properties of argumentative entailment.

If $\mathcal{O}$ is inconsistent and there exists an axiom ϕ such that $\mathcal{O} \not\models_p \phi$ where $\models_p$ is an entailment relation, then we say $\models_p$ is *paraconsistent*. It is well known that classical entailment $\models$ is not paraconsistent.

Theorem 1 *The a-entailment $\models_a$ is paraconsistent.*

In Example 1, $\mathcal{O}_1 \not\models_a Fly(tweety)$ and $\mathcal{O}_1 \not\models_a Penguin \sqsubseteq Fly$.

Most existing semantics for paraconsistent reasoning in DLs are much weaker than the classical semantics in this sense that there exists a consistent ontology $\mathcal{O}$ and an axiom ϕ such that $\mathcal{O} \models \phi$ (also called *consistency preservation*) but ϕ is not entailed by $\mathcal{O}$ under the paraconsistent semantics. The following result shows that the a-entailment does not have such shortcoming.

The following result shows that the a-entailment is consistency-preserving.

Theorem 2 *Let $\mathcal{O}$ be a consistent and coherent ontology and ϕ an axiom. $\mathcal{O} \models_a \phi$ if and only if $\mathcal{O} \models \phi$.*

Proof Because $\mathcal{O}$ is consistent and coherent, $\mathcal{O} \models \phi \Leftrightarrow$ the argumentation framework $\langle \mathcal{P}, \mathcal{C} \rangle$ for ϕ whose each argument tree $\mathcal{P} \neq \emptyset$ has exactly one node and $\mathcal{C}$ is the empty set by Proposition 9 $\Leftrightarrow$ there exists a successful argument tree for ϕ because it contains only one node $\Leftrightarrow \mathcal{O} \models_a \phi$. □

Theorem 2 ensures that the a-entailment over consistent ontologies preserves the classical entailment. That is, our semantics is as the same as classical semantics in dealing with consistent and coherent ontologies.

An entailment relation $\models_m$ is *monotonic* if $\mathcal{O}' \models_m \phi$ implies $\mathcal{O} \models_m \phi$ for any $\mathcal{O}' \subseteq \mathcal{O}$; *nonmonotonic* otherwise. Another characteristic property of $\models_a$ is its nonmonotonic nature.

Theorem 3 *The a-entailment $\models_a$ is nonmonotonic.*

For instance, let $\mathcal{O} = (\{A \sqsubseteq B\}, \{A(a)\})$ and $\mathcal{O}' = (\{A \sqsubseteq B\}, \{A(a), \neg B(a)\})$. Obviously, $\mathcal{O} \subseteq \mathcal{O}'$ and $\mathcal{O} \models_a B(a)$ while $\mathcal{O}' \not\models_a B(a)$.

While the argumentative semantics is nonmonotonic in general, it possesses a kind of cautious monotonicity, which is usually referred to as *splitting property* (see Arieli et al. 2011).

For instance, let $\mathcal{O} = (\{\top \sqsubseteq A,\ A \sqsubseteq \forall P.B,\ \forall P.B \sqsubseteq \bot,\ \forall R_1.C \sqsubseteq \exists R_2.D\}, \{A(a), \forall R_1.C(b)\})$ be an ontology, $\mathcal{O}$ can be split into $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_2$ where $\mathcal{O}_1 = (\{\forall R_1.C \sqsubseteq \forall R_2.D\}, \{\forall R_1.C(b)\})$ and $\mathcal{O}_2 = (\{\top \sqsubseteq A, A \sqsubseteq \forall P.B, \forall P.B \sqsubseteq \bot\}, \{A(a)\})$. Then, let $\phi = \forall R_1.C(b)$, the problem of deciding $\mathcal{O} \models_{d,f} \phi$ can be reduced to checking whether $\mathcal{O}_1 \models_{d,f} \phi$. Notice that $\mathcal{O}_1$ is consistent and $Sig(\mathcal{O}_1) \cap Sig(\mathcal{O}_2) = \emptyset$ where $Sig(\mathcal{O})$ is a set of all symbols occurring in $\mathcal{O}$. For some non-monotonic semantics, these two conditions are sufficient to guarantee the validity of the splitting property. However, it is not the case when $\mathcal{O}_1$ contains more than one axiom.

Formally, we say $\mathcal{O}$ is split into $\mathcal{O}'$ and $\mathcal{O}''$, denoted by $\mathcal{O} = \mathcal{O}' \oplus \mathcal{O}''$, if (1) $\mathcal{O} = \mathcal{O}' \cup \mathcal{O}''$, and (2) $Sig(\mathcal{O}') \cap Sig(\mathcal{O}'') = \emptyset$.

Theorem 4 *Let $\mathcal{O} = \mathcal{O}' \oplus \mathcal{O}''$ where $\mathcal{O}'$ is consistent and coherent. If $\mathcal{O}' \models \phi$ then $\mathcal{O} \models_a \phi$ for each axiom ϕ such that $Sig(\phi) \cap Sig(\mathcal{O}'') = \emptyset$.*

Proof To prove that $\mathcal{O} \models_a \phi$, we only need to show that there exists a successful argument tree for ϕ in $\mathcal{O}$. If $\mathcal{O}'$ is consistent and coherent and $\mathcal{O}' \models \phi$, then the argumentation framework $\mathcal{F}$ of ϕ w.r.t. $\mathcal{O}'$ is in form of $\langle \mathcal{P}, \mathcal{C} \rangle$ where all argument trees for ϕ in $\mathcal{P}$ has exactly one node and $\mathcal{C}$ is empty by Proposition 9. Because

$Sig(\phi) \cap Sig(\mathcal{O}'') = \emptyset$, all argument trees for/against ϕ must not contain any symbol in $Sig(\mathcal{O}'')$. Then $\mathcal{F}$ is also the argumentation framework of ϕ w.r.t. $\mathcal{O}$. That is, there exists at least a successful argument tree for ϕ in $\mathcal{O}$. □

One advantage of the splitting property is that the paraconsistent reasoning in ontology $\mathcal{O}$ can be localized into the classical reasoning in a consistent module of $\mathcal{O}$, which is usually smaller than the original $\mathcal{O}$. Such a property can be very useful for a highly distributed ontology system.

Moreover, our semantics inherits a unique property called *justifiability* of argumentation theory (Dung 1995a; Besnard and Hunter 2001). Formally, an axiom ϕ is *justifiable* in an ontology $\mathcal{O}$ if and only if there exists a self-protected argument for ϕ but there exists no self-protected argument against ϕ.

All justifiable axioms in a given ontology can be characterized by the a-entailment.

Proposition 10 *Let $\mathcal{O}$ be an ontology and ϕ an axiom. ϕ is justifiable in $\mathcal{O}$ if and only if $\mathcal{O} \models_a \phi$ and $\mathcal{O} \not\models_a \sim \phi$.*

Proof ϕ is justifiable in $\mathcal{O}$ $\Leftrightarrow$ $\mathcal{O} \models_a \phi$ and $\mathcal{O} \not\models_a \sim \phi$ $\Leftrightarrow$ ϕ is accepted and $\sim \phi$ is not accepted in $\mathcal{O}$ $\Leftrightarrow$ there exists a self-protected argument for ϕ but there exists no self-protected argument against ϕ. □

In Example 1, $\neg Fly(tweety)$ is justifiable in $\mathcal{O}_1$. Neither $A(a)$ nor $B(a)$ is justifiable in the ontology $(\{A \sqsubseteq B\}, \{A(a), \neg B(a)\})$.

A semantics is *multi-valued* if the number of its truth values is greater than 2 (“true” and “false”) (Arieli et al. 2011).

To show that our semantics is multi-valued, we need to quantify binary argumentation framework by introducing two quantification functions: *categorizer* and *accumulator*.

The *categorizer* is a function, denoted by $\mathbf{c}$, from the set of argument trees to $\{0, 1\}$ such that $\mathbf{c}(T) = 1$ if and only if argument tree T is successful. In Fig. 1, $\mathbf{c}(T_1) = \mathbf{c}(T_4) = \mathbf{c}(T_6) = 0$ and $\mathbf{c}(T_2) = \mathbf{c}(T_3) = \mathbf{c}(T_5) = 1$.

The *categorization* of a set of argument trees is the collection of their categorizer values. The *accumulator* of an axiom ϕ is a function, denoted by $\mathbf{a}$, from categorizations to the set $\{(1, 1), (1, 0), (0, 1), (0, 0)\}$. Let $\langle X, Y\rangle$ be a categorization of argumentation framework for an axiom ϕ, then $\mathbf{a}(\langle X, Y\rangle) = (w(X), w(Y))$ where $w(Z) = 1$ if and only if $1 \in Z$.

Let $\mathcal{O}$ be an ontology and ϕ an axiom. If $\langle X, Y\rangle$ is a categorization of argumentation framework for ϕ, then $\mathbf{a}_{\mathcal{O}}(\phi) = (w(X), w(Y))$ where $w(Z) = 1$ if and only if $1 \in Z$. If it is clear in the context, we abbreviate $\mathbf{a}_{\mathcal{O}}(\phi)$ as $\mathbf{a}(\phi)$.

In Example 1, $\mathbf{a}(Fly(tweety)) = (0, 1)$ since T_2, T_3 are successful but T_1 is not successful. $\mathbf{a}(Penguin \sqsubseteq Fly) = (0, 1)$ since T_5 is successful but T_4 is not successful.

The following proposition shows the relationship between accumulator and $\sim$.

Proposition 11 *Let $\mathcal{O}$ be an ontology. Assume that $\mathbf{a}(\phi) = (i, j)$ where $i, j \in \{0, 1\}$. We have*

1. *if $i + j = 1$ then $\mathbf{a}(\sim \phi) = (j, i)$;*
2. *if $i + j = 0$ or 2 then $\mathbf{a}(\sim \phi) = (i, j)$.*

Proof It directly follows from Definition 7, the definitions of consistency-negation, coherency-negation and the definition of accumulator. □

Proposition 11 provides a theoretical support for our repair operators presented in the next section.

Based on accumulator, we then define a valuation function.

Definition 9 Let $\mathcal{O}$ be an ontology and ϕ an axiom. The *argumentative valuation* (*a-valuation*, for short) is a function, denoted by $\mathbf{v}_a$, from a set of axioms and assertions to a set of four values {t,f,B,U}, defined as follows:

$$\mathbf{v}_a(\phi) = \begin{cases} \text{B, if} & \mathbf{a}(\langle X, Y\rangle) = (1, 1); \\ \text{t, if} & \mathbf{a}(\langle X, Y\rangle) = (1, 0); \\ \text{f, if} & \mathbf{a}(\langle X, Y\rangle) = (0, 1); \\ \text{U, if} & \mathbf{a}(\langle X, Y\rangle) = (0, 0). \end{cases}$$

where $\langle X, Y\rangle$ is a categorization of the argumentation framework of ϕ in $\mathcal{O}$.

Intuitively speaking,

1. $\mathbf{v}_a(\phi) = B$ means that there exists a successful argument tree for ϕ and a successful argument tree for $\sim \phi$;
2. $\mathbf{v}_a(\phi) = t$ means that there exists a successful argument tree for ϕ but no successful argument tree for $\sim \phi$;
3. $\mathbf{v}_a(\phi) = f$ means that there exists a successful argument tree for ϕ but no successful argument tree for $\sim \phi$;
4. $\mathbf{v}_a(\phi) = U$ means that there exists neither any successful argument tree for ϕ nor any successful argument tree for $\sim \phi$.

The next result shows that there exists a close relation between the a-entailment ($\models_a$) and the argumentative valuation ($\mathbf{v}_a$).

Theorem 5 *Let $\mathcal{O}$ be an ontology and ϕ an axiom. Then the following propositions are equivalent to each other:*

1. $\mathcal{O} \models_a \phi$;
2. $\mathbf{v}_a(\phi) \in \{B,t\}$;
3. *there exists an argument tree T for ϕ such that* $\mathbf{c}(T) = 1$.

Proof $\mathcal{O} \models_a \phi \Leftrightarrow$ there exists a successful tree T of ϕ in $\mathcal{O}$ by Definition 8 $\Leftrightarrow \mathbf{c}(T) = 1$ by the definitions of accumulators $\Leftrightarrow \mathbf{v}(\phi) \in \{B, t\}$ by the definition of categorizer and valuation function. □

In other words, our semantics is still four-valued.

For instance, let $\mathcal{O} = (\{A \sqsubseteq B\}, \{A(a), \neg B(a)\})$ be an ontology, $\mathbf{v}(A(a)) = \mathbf{v}(B(a)) = U$, that is, the answers to (query) $A(a)$ and $B(a)$ are unknown. Intuitively, neither $A(a)$ nor $\neg A(a)$ is true in the ontology.

However, $\models_a$ does not satisfy the TBox-*preserved* property, that is, let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology, it does not always hold that $\mathcal{O} \models_a \phi$ if and only if $\mathcal{O} \models_a \phi$. For instance, in Example 1, $\mathcal{O}_1 \models_a$ *Penguin* $\sqsubseteq \neg$*Fly* while $\mathcal{T}_1 \not\models_a$ *Penguin* $\sqsubseteq \neg$*Fly*. The reason is that all axioms are impartially treated so that axioms in the ABox still affect reasoning with TBoxes.

In summary, instead of roughly either discarding or ignoring contradictory information in existing approaches to handling inconsistent ontologies, our scenario based on argumentation framework cautiously treats contradictory information by analyzing the internal relation between its semantics and the whole semantics of a given ontology and then provides more reasonable answers.

5 Repairing ontologies with conflicts by using argumentation

Ontology engineers generally think that conflicts should be removed to maintain consistency of ontologies. As a practical application of the Semantic Web, maintaining consistency is an important task in ontology management. Because an ontology contains two parts: TBox (collection of inner knowledge) and ABox (collection of outer knowledge), there are three kinds of conflicts occurring in ontologies: the first only occurring in TBox; the second only occurring in ABox and the third occurring between TBox and ABox. Note that the second could be taken as the third with empty TBox. There are some proposals to repair ontologies with the first (Kalyanpur et al. 2006b; Meyer et al. 2006) and the third (Du and Shen 2008). The common idea of them is based on maximal consistent or coherent subsets of ontologies in syntax. In this section, we develop a novel semantic-based scenario to eliminate conflicts by employing our argumentative entailment presented in the previous section. We consider two kinds of conflicts (i.e., inconsistency and incoherency) occurring in ontologies. Indeed, incoherency not only is taken as a kind of errors but also causes inconsistency (see Kalyanpur et al. 2006b).

For instance, in Example 1, both $\mathcal{T}_1$ and $\mathcal{A}_1$ are consistent while the inconsistency of $\mathcal{O}_1$ is caused by the incoherency of $\mathcal{T}_1$ since *Penguin* is an unsatisfiable concept name in $\mathcal{T}_1$.

Inconsistency or incoherency of an ontology might be mainly caused by its over-definition in ontology engineering. For instance, it clearly shows that two axioms $A \sqsubseteq B$ and $A \sqsubseteq \neg B$ for defining concept A bring incoherence. A key idea of repairing inconsistent or incoherent ontologies is removing redundant axioms.

In this section, we develop several novel operators based on argumentation framework to repair incoherent or inconsistent ontologies.

5.1 Repairing description logic ontologies using argumentation

As discussed previously, there are four accumulators of our argumentation framework for each axiom in a TBox. Note that our candidate repair operator is used to maintain coherency of a TBox while our argumentative entailment is used to tolerate conflicts. Because of this, our candidate operator needs more cautious consideration.

Definition 10 Let $\mathcal{O}$ be an ontology and $\mathcal{S}$ a set of axioms. An *argumentation-based repair operator* (*a-repair operator*, for short) $\triangle_a$ is defined as follows:

$$\triangle_a(\mathcal{O},\mathcal{S}) = \{\phi \in \mathcal{S} \mid \mathbf{a}_{\mathcal{O}}(\phi) = (1,0)\}.$$

Intuitively speaking, for each axiom ϕ of $\triangle_a(\mathcal{S}, \mathcal{O})$, the a-valuation is "$t$" (true), that is, there exists a successful argument tree for ϕ but no successful argument tree for $\sim\phi$ in $\mathcal{O}$. We directly use $\triangle_a(\mathcal{S})$ to denote $\triangle_a(\mathcal{S}, \mathcal{S})$.

In constructing repaired ontologies, we follow from two facts: (1) incoherence or inconsistency of a TBox is mainly caused by itself; (2) inconsistency of an ABox w.r.t. a coherent and consistent TBox is mainly caused by itself. Based on the above two facts, we define a repaired ontology as follows.

Definition 11 Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology. An *argumentatively repaired* (*a-repaired*) ontology of $\mathcal{O}$, denoted by $\triangle_a(\mathcal{O})$, is defined as follows:

$$\triangle_a(\mathcal{O}) = (\triangle_a(\mathcal{T}), \triangle_a(\mathcal{O}^a, \mathcal{A})).$$

where $\mathcal{O}^a = (\triangle_a(\mathcal{T}), \mathcal{A})$.

Intuitively, the repair contains two processes: the first is repairing the TBox and the second is repairing the ABox w.r.t. the repaired TBox.

Note that $\triangle_a(\mathcal{O})$ is always finite if $\mathcal{O}$ is finite. That is, this a-repair operator is well defined. Moreover, Definition 11 provides one of the methods to repair inconsistent or incoherent ontology via a-repair operator. In other words, we could develop several methods of repairing by using a-repair operator. In addition, a-repaired ontologies preserve the syntactic structure of their original ontologies.

The repair can distinguish the inconsistency caused by the incoherency of TBoxes from other inconsistencies.

In Example 1, $\triangle_a(\mathcal{T}) = \{Bird \sqsubseteq Fly,\ DegradedBird \sqsubseteq \neg Fly,\ Swallow \sqsubseteq Bird,\ Penguin \sqsubseteq \exists hasFood.Fish,\ Swallow \sqsubseteq \neg Penguin\}$ and $\triangle_a(\mathcal{O}_a, \mathcal{A}_1) = \mathcal{A}_1$. We find that the coherency of $\triangle_a(\mathcal{T})$ is obtained by removing two axioms: $Penguin \sqsubseteq Bird$ and $Penguin \sqsubseteq \neg Fly$ which cause the incoherency of $\mathcal{T}_1$ together with $Bird \sqsubseteq Fly$.

Example 2 Let $\mathcal{O}_2 = (\mathcal{T}_2, \mathcal{A}_2)$ where TBox $\mathcal{T} = \{A_1 \sqsubseteq A_2,\ A_2 \sqsubseteq A_4,\ A_1 \sqsubseteq A_3,\ A_3 \sqsubseteq A_4,\ A_1 \sqsubseteq \neg A_4\}$ and $\mathcal{A}_4 = \{A_1(a), A_3(a), \neg A_4(a), \neg A_4(b)\}$. Then $\triangle_a(\mathcal{O}_2) = (\triangle_a(\mathcal{T}_2), \triangle_a(\mathcal{O}_2^a, \mathcal{A}_4))$ where $\triangle_a(\mathcal{T}_4) = \{A_1 \sqsubseteq A_2,\ A_2 \sqsubseteq A_4,\ A_1 \sqsubseteq A_3,\ A_3 \sqsubseteq A_4\}$ and $\triangle_a(\mathcal{O}_2^a, \mathcal{A}_2) = \{A_1(a), A_3(a), \neg A_4(b)\}$. Note that the axiom $A_1 \sqsubseteq \neg A_4$ and the assertion $\neg A_4(a)$ are absent in the a-repaired ontology. It easily shows that $\triangle_a(\mathcal{O}_2)$ is coherent and consistent.

The a-repair operator has some good properties.

Theorem 6 *Let $\mathcal{O}$ be an ontology. We have*

1. *$\triangle_a(\mathcal{O})$ is coherent and consistent;*
2. *$\mathcal{O} = \triangle_a(\mathcal{O})$ if $\mathcal{O}$ is consistent and coherent;*
3. *if $\mathcal{O} = \mathcal{O}' \cup \mathcal{O}''$ and $Sig(\mathcal{O}') \cap Sig(\mathcal{O}'') = \emptyset$, then $\triangle_a(\mathcal{O}) = \triangle_a(\mathcal{O}') \cup \triangle_a(\mathcal{O}'')$.*

Proof

1. Based on the definition of accumulator and Definition 11, for any axiom $\phi \in \triangle_a(\mathcal{O})$, we have $\sim\phi \in \triangle_a(\mathcal{O})$. Therefore, $\triangle_a(\mathcal{O})$ is coherent and consistent.

2. If $\mathcal{O}$ is consistent and coherent, then the argumentation framework of each axiom has the form of $\langle \mathcal{P}, \mathcal{C} \rangle$ where $\mathcal{P}$ contains only one node and $\mathcal{C}$ is empty by Proposition 8. That is $\mathbf{a}(\phi) = (1, 0)$ for any axiom $\phi \in \mathcal{O}$. Thus $\mathcal{O} = \triangle_a(\mathcal{O})$.
3. If $Sig(\mathcal{O}') \cap Sig(\mathcal{O}'') = \emptyset$, then $Sig(\triangle_a(\mathcal{O}')) \cap Sig(\triangle_a(\mathcal{O}'')) = \emptyset$ since $\triangle_a(\mathcal{O}') \subseteq \mathcal{O}'$ and $\triangle_a(\mathcal{O}'') \subseteq \mathcal{O}''$. Thus $\triangle_a(\mathcal{O}) = \{\phi \in \mathcal{O} \mid \mathbf{a}(\phi) = (1, 0)\} = \{\phi \in \mathcal{O}' \cup \mathcal{O}'' \mid \mathbf{a}(\phi) = (1, 0)\} = \{\phi \in \mathcal{O}' \mid \mathbf{a}(\phi) = (1, 0)\} \cup \{\phi \in \mathcal{O}'' \mid \mathbf{a}(\phi) = (1, 0)\} = \triangle_a(\mathcal{O}') \cup \triangle_a(\mathcal{O}'')$. □

In Theorem 6, the first item states that our repair operator $\triangle_a$ could maintain consistency and coherency of ontologies; the second shows that the a-repaired ontology of a coherent or consistent ontology is itself; the third tells that a-repair operator satisfies the splitting property. Taking advantage of the splitting property, we could realize modularized management of ontologies.

In addition, for any ontology, its a-repaired ontology is unique. In Example 2, the result after repairing is only $\triangle_a(\mathcal{O}_2)$.

In short, for an inconsistent or incoherent ontology, its a-repaired ontology is a reasonable and suitable alternative one since all information which might conflict with others is excluded from such an alternative ontology. Our repairing obeys the important principle of justifiability compared with existing proposals of ontology repair.

5.2 Approximatively repairing description logic ontologies using argumentation

Though the results of repairing ontologies based on our operator are justifiable by removing redundant axioms from ontologies, some of those redundant axioms are still valuable sometimes.

Example 3 Let $\mathcal{O}_3 = (\mathcal{T}_3, \mathcal{A}_3)$ where $\mathcal{T}_3 = \{A \sqsubseteq B, A \sqsubseteq \neg B\}$ and $\mathcal{A}_3 = \{A(a), \neg B(a), A(b), B(b)\}$. We have $\triangle_a(\mathcal{T}_3) = \emptyset$. Though there is no more evidence which is more reliable than axiom $A \sqsubseteq B$ and axiom $A \sqsubseteq \neg B$, the candidate result either $\{A \sqsubseteq B\}$ or $\{A \sqsubseteq B\}$ might be sometimes better than $\triangle_a(\mathcal{T}_3)$ which could not provide any information.

Next, we develop two approximative operators to obtain more meaningful results by relaxing some restrictions of a-repair operator.

Definition 12 Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology. Two *approximative a-repair operators* $\triangle_a^1$ and $\triangle_a^2$ are defined as follows:

1. $\triangle_a^1(\mathcal{O}, \mathcal{S}) = \{\phi \in \mathcal{S} \mid \mathbf{a}_{\mathcal{O}}(\phi) = (1, 0)\} \cup \{\sim \phi \in \mathcal{S} \mid \mathbf{a}_{\mathcal{O}}(\phi) = (0, 1)\}$.
2. $\triangle_a^2(\mathcal{O}, \mathcal{S}) = \{\phi \in \mathcal{S} \mid \mathbf{a}_{\mathcal{O}}(\phi) = (1, 0)\}$ by adding $\phi \in \mathcal{S}$ with $\mathbf{a}_{\mathcal{O}}(\phi) = (0, 0)$ in the following way:
   - if $\phi, \sim \phi \in \mathcal{S}$, then one of ϕ and $\sim \phi$ is added;
   - otherwise, ϕ is directly added.

$\triangle_a^1(\mathcal{O})$ and $\triangle_a^2(\mathcal{O})$ are analogously defined as in Definition 11.

Intuitively, two new operators $\triangle_a^1$ and $\triangle_a^2$ could be taken as extensions of the a-repair operator $\triangle_a$ via two selection strategies. The operator $\triangle_a^1$ could be used to capture those axioms which are testified to be justifiable and the operator $\triangle_a^2$ could be used to capture the maximality of consistency and coherency.

Corresponding to the approximative a-repair operators $\triangle_a^1$ and $\triangle_a^2$, we say operator $\triangle_a$ the *normal repair operator*.

Theorem 7 *Let $\mathcal{O}$ be an ontology. We have*

1. *$\triangle_a^1(\mathcal{O})$ and $\triangle_a^2(\mathcal{O})$ are consistent and coherent;*
2. *if $\mathcal{O}$ is consistent and coherent, then $\triangle_a^1(\mathcal{O}) = \triangle_a^2(\mathcal{O}) = \mathcal{O}$;*
3. *if $\mathcal{O} = \mathcal{O}' \oplus \mathcal{O}''$ and $Sig(\mathcal{O}') \cap Sig(\mathcal{O}'') = \emptyset$, $\triangle_a^1(\mathcal{O}) = \triangle_a^1(\mathcal{O}') \oplus \triangle_a^1(\mathcal{O}'')$ and $\triangle_a^2(\mathcal{O}) = \triangle_a^2(\mathcal{O}') \oplus \triangle_a^2(\mathcal{O}'')$ where $\mathcal{O}' = (\mathcal{T}', \mathcal{A}')$, $\mathcal{O}'' = (\mathcal{T}'', \mathcal{A}'')$ and $(\mathcal{T}', \mathcal{A}') \oplus (\mathcal{T}'', \mathcal{A}'') = (\mathcal{T}' \cup \mathcal{T}'', \mathcal{A}' \cup \mathcal{A}'')$.*

Proof This proof is analogous to the proof of Theorem 6. □

In the following proposition, we discuss the relationships between two approximative operators ($\triangle_a^1$ and $\triangle_a^2$) and a-repair operator ($\triangle_a$).

Proposition 12 *Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology. We have (1) $\triangle_a(\mathcal{O}) \subseteq \triangle_a^1(\mathcal{O})$; and (2) $\triangle_a(\mathcal{O}) \subseteq \triangle_a^2(\mathcal{O}) \subseteq \mathcal{O}$.*

Note that the results by using operators $\triangle_a$ and $\triangle_a^1$ are unique while the results by using the operator $\triangle_a^2$ are often multiple.

In Example 3, $\triangle_a^2(\mathcal{T})$ has two possible results as follows: $\{A \sqsubseteq B\}$ or $\{A \sqsubseteq \neg B\}$. Then $\triangle_a^2(\mathcal{O})$ has four possible results as follows: $(\{A \sqsubseteq B\}, \{A(a), A(b), B(b)\})$, $(\{A \sqsubseteq B\}, \{\neg B(a), A(b), B(b)\})$, $(\{A \sqsubseteq \neg B\}, \{A(a), \neg B(a), A(b)\})$ and $(\{A \sqsubseteq \neg B\}, \{A(a), \neg B(a), B(b)\})$.

Besides, those repaired ontologies by using a-repair operators possibly contain some existence axioms in form of $\exists(C \sqcap \neg D)$ (see Horrocks and Patel-Schneider 2003) because of the consistency-negation of concept inclusions in form of $C \sqsubseteq D$. Technically, we argue that it is feasible that the *forgetting* and *uniform interpolation* (Konev et al. 2009) can be employed to recover $C \sqsubseteq E$ (if necessary) when $D \sqsubseteq E$ is removed from $\{C \sqsubseteq D, D \sqsubseteq E\}$.

6 Examples: two practical ontologies

Two popular ontologies, namely, *badFood* ontology and *buggyPolicy* ontology (Kalyanpur et al. 2006a), contain unsatisfiable concepts which are caused by modeling error and overdefining respectively. However, the problems of finding all unsatisfiable concepts and those errors (axioms) which cause unsatisfiability of DL concepts in a given ontology are still open.

badFood ontology The badFood ontology, written by $\mathcal{O}_b$, contains 26 axioms φ_i as follows:

$$\begin{aligned}
\varphi_1 &: & \exists eats.\top &\sqsubseteq Person\\
\varphi_2 &: & \top &\sqsubseteq \forall eats.Food\\
\varphi_3 &: & LactoVegetarian &\sqsubseteq \forall eats.LactoVegetarianFood\\
\varphi_4 &: & Omnivore &\equiv Person \sqcap (\exists eats.Meat)\\
\varphi_5 &: & OvoLactoVegetarian &\sqsubseteq \forall eats.OvoLactoVegetarianFood\\
\varphi_6 &: & OvoVegetarian &\sqsubseteq \forall eats.OvoVegetarianFood\\
\varphi_7 &: & Vegan &\sqsubseteq \forall eats.VeganFood\\
\varphi_8 &: & Vegetarian &\sqsubseteq \forall eats.VegetarianFood\\
\varphi_9 &: & Dairy &\sqsubseteq \neg Eggs\\
\varphi_{10} &: & DairyandEggs &\equiv Dairy \sqcup Eggs\\
\varphi_{11} &: & OvoVegetarianFood &\equiv Food \sqcap (\neg(Dairy \sqcup Meat))\\
\varphi_{12} &: & VeganFood &\equiv Food \sqcap (\neg(DairyandEggs \sqcup Meat))\\
\varphi_{13} &: & Seafood &\sqsubseteq Meat\\
\varphi_{14} &: & Seafood &\sqsubseteq \neg VegetarianFood\\
\varphi_{15} &: & Meat &\sqsubseteq Food\\
\varphi_{16} &: & VegetarianFood &\equiv Food \sqcap \neg Meat\\
\varphi_{17} &: & LactoVegetarianFood &\sqsubseteq VegetarianFood\\
\varphi_{18} &: & LactoVegetarianFood &\sqsubseteq \neg OvoVegetarianFood\\
\varphi_{19} &: & OvoLactoVegetarianFood &\sqsubseteq LactoVegetarianFood\\
\varphi_{20} &: & OvoLactoVegetarian &\sqsubseteq Vegetarian\\
\varphi_{21} &: & LactoVegetarian &\sqsubseteq OvoLactoVegetarian\\
\varphi_{22} &: & OvoVegetarian &\sqsubseteq OvoLactoVegetarian\\
\varphi_{23} &: & Vegan &\sqsubseteq Vegetarian\\
\varphi_{24} &: & Vegetarian &\sqsubseteq Person\\
\varphi_{25} &: & OvoVegetarianFood &\sqsubseteq VegetarianFood\\
\varphi_{26} &: & OvoVegetarianFood &\sqsubseteq OvoLactoVegetarianFood
\end{aligned}$$

Note that $OvoVegetarian \sqsubseteq \forall eats.\neg OvoVegetarianFood$, which conflicts φ_6, can be inferred from $\{\varphi_5, \varphi_{18}, \varphi_{22}\}$. Thus the concept $OvoVegetarian$ is unsatisfiable. Then the badFood ontology is incoherent which unavoidably causes the inconsistency together with some assertions (external knowledge).

We add axioms $\varphi_{27} = OvoVegetarian(person)$, $\varphi_{28} = eats(person, food)$ where *preson* and *food* are two individuals, in $\mathcal{O}_b$ and then we obtain a new ontology denoted by $\mathcal{O}_b^*$. It is not hard to conclude that $\mathcal{O}_b^*$ is inconsistent because $OvoVegetarianFood(food)$ and $\neg OvoVegetarianFood(food)$ conflict with each other.

Table 1 Reasoning results over badFood ontology

Axiom: ϕ / $\sim\phi$	$(\models_4, \mapsto)$	$(\models_4, \sqsubset)$	$(\models_4, \rightarrow)$	$\models_Q$	$\models_c$	$\models_b$	$\models_a$
$\varphi_5/\sim\varphi_5$	yes/no	yes/no	yes/yes	yes/no	no/no	yes/no	yes/no
$\varphi_6/\sim\varphi_6$	yes/no	yes/yes	yes/yes	yes/no	no/no	yes/no	yes/yes
$\varphi_{18}/\sim\varphi_{18}$	yes/no	yes/yes	yes/yes	yes/no	no/no	yes/no	yes/no
$\varphi_{19}/\sim\varphi_{19}$	yes/no	yes/no	yes/yes	yes/no	no/no	yes/no	yes/no
$\varphi_{22}/\sim\varphi_{22}$	yes/no	yes/no	yes/yes	yes/no	no/no	yes/no	yes/no
$\varphi_{26}/\sim\varphi_{26}$	yes/no	yes/no	yes/yes	yes/no	yes/no	yes/no	yes/no
$\varphi_{27}/\sim\varphi_{27}$	yes/no	yes/no	yes/yes	yes/no	no/no	yes/yes	no/no
$\alpha_1/\sim\alpha_1$	no/no	yes/yes	yes/yes	yes/ yes	no/no	yes/yes	no/yes
$\alpha_2/\sim\alpha_2$	no/no	yes/no	yes/yes	yes/no	no/no	yes/yes	yes/no
$\alpha_3/\sim\alpha_3$	no/no	yes/no	yes/yes	yes/no	no/no	yes/yes	yes/no
$\alpha_4/\sim\alpha_4$	yes/no	yes/no	yes/yes	yes/no	yes/no	yes/no	yes/no
$\alpha_5/\sim\alpha_5$	yes/no	yes/no	yes/yes	yes/no	yes/no	yes/no	yes/no
$\alpha_6/\sim\alpha_6$	no/no	yes/no	yes/yes	yes/no	no/no	yes/yes	yes/no

Now, we additionally consider some interesting queries as follows:

$$\begin{aligned}
&\alpha_1 : OvoVegetarianFood(food)\\
&\alpha_2 : LactoVegetarianFood(food)\\
&\alpha_3 : OvoLactoVegetarianFood(food)\\
&\alpha_4 : VegetarianFood(food)\\
&\alpha_5 : Food(food)\\
&\alpha_6 : OvoLactoVegetarian(person)
\end{aligned}$$

Note that under non-classical semantics, though the answer of a query is either "yes" or "no", it does not mean that its negation has the contrary answer since it is possibly assigned as other value (e.g., "unknown") besides "true" or "false". To refine answers, we also consider a query and its negation (α / $\sim\alpha$) together. The experimental results are shown in Table 1.

Notations:

- $(\models_4, \mapsto)$: four-valued entailment under material implication (Ma et al. 2007);
- $(\models_4, \sqsubset)$: four-valued entailment under internal implication (Ma et al. 2007);
- $(\models_4, \rightarrow)$: four-valued entailment under strong implication (Ma et al. 2007);
- $\models_Q$: quasi-classical entailment (Zhang et al. 2009; Zhang and Lin 2012);
- $\models_c$: cautious entailment: $\mathcal{O} \models_c \varphi$ if for each maximal consistent sub-ontology $\mathcal{O}'$ of $\mathcal{O}$, we have $\mathcal{O}' \models \varphi$ (Huang et al. 2005);
- $\models_b$: brave entailment: $\mathcal{O} \models_b \varphi$ if there exists some maximal consistent sub-ontology $\mathcal{O}'$ of $\mathcal{O}$ such that $\mathcal{O}' \models \varphi$ (Huang et al. 2005).

Moreover, we can also compute the repaired ontologies of the badFood ontology by using argumentation as follows: $\triangle_a(\mathcal{O}_b^*) = \mathcal{O}_b^* - \{\varphi_6, \varphi_{27}\}$; $\triangle_a^1(\mathcal{O}_b) = \triangle_a(\mathcal{O}_b^*)$ and $\triangle_a^2(\mathcal{O}_b) = \triangle_a^1(\mathcal{O}_b^*) = \triangle_a(\mathcal{O}_b^*) \cup \{\sim\varphi_{27}\}$.

As a result, *OvoVegetarianFood* is satisfiable and these repaired ontologies are coherent and consistent.

buggyPolicy ontology The buggyPolicy ontology, written by $\mathcal{O}_p$, contains 19 axioms ψ_i as follows:

$$\begin{aligned}
\psi_1 &: \exists testProperty.\top \sqsubseteq Reliable\\
\psi_2 &: ExactlyOneExamplePolicy \sqsubseteq Policy\\
\psi_3 &: GeneralReliabilityKerberosPolicy \sqsubseteq Policy\\
\psi_4 &: GeneralReliabilityUserPolicy \equiv Reliable \sqcap UserToken\\
\psi_5 &: GeneralReliabilityUserPolicy \sqsubseteq Policy\\
\psi_6 &: GeneralReliabilityUserPolicy \sqsubseteq \neg Messaging\\
\psi_7 &: X509 \sqsubseteq SecurityTokenType\\
\psi_8 &: IncoherentPolicy \sqsubseteq Policy\\
\psi_9 &: Kerberos \sqsubseteq SecurityTokenType\\
\psi_{10} &: Kerberos \sqsubseteq \neg Messaging\\
\psi_{11} &: Reliable \sqsubseteq Messaging\\
\psi_{12} &: RetryOnFailureUserPolicy \sqsubseteq Policy\\
\psi_{13} &: RetryUntilSucceedUserPolicy \sqsubseteq Policy\\
\psi_{14} &: RetryOnFailure \sqsubseteq Reliable\\
\psi_{15} &: RetryUntilSucceed \sqsubseteq Reliable\\
\psi_{16} &: UserToken \sqsubseteq SecurityTokenType\\
\psi_{17} &: RetryOnFailureUserPolicy \equiv RetryOnFailure \sqcap UserToken\\
\psi_{18} &: RetryUntilSucceedUserPolicy \equiv RetryUntilSucceed \sqcap UserToken\\
\psi_{19} &: IncoherentPolicy \equiv RetryOnFailureUserPolicy \sqcap\\
&\qquad RetryUntilSucceedUserPolicy
\end{aligned}$$

Note that $GeneralReliabilityUserPolicy \sqsubseteq Messaging$ can be inferred from $\{\psi_4, \psi_{11}\}$, which conflicts with ψ_6. Thus $GeneralReliabilityUserPolicy$ is unsatisfiable.

Now, we add two axioms $\psi_{20} = GeneralReliabilityUserPolicy(id)$ and $\psi_{21} = IncoherentPolicy(id)$, where id is an individual, in $\mathcal{O}_p$ and then we obtain a new ontology denoted by $\mathcal{O}_p^*$. It is not hard to conclude that $\mathcal{O}_p^*$ is inconsistent because $Messaging(id)$ and $\neg Messaging(id)$ conflict with each other.

Now, we additionally consider some interesting queries and experimental results are shown in Table 2.

$$\beta_1 : Messaging(id) \quad \beta_2 : Reliable(id)$$
$$\beta_3 : Policy(id) \quad \beta_4 : RetryUntilSucceed(id)$$

Besides, we can also compute the repaired ontologies of the buggyPolicy ontology by using argumentation as follows: $\triangle_a(\mathcal{O}_p^*) = \mathcal{O}_p - \{\psi_5, \psi_{20}\}$; $\triangle_a^1(\mathcal{O}_p^*) = \triangle_a(\mathcal{O}_p^*) \cup \{\sim \psi_5\}$ and $\triangle_a^2(\mathcal{O}_p^*) = \triangle_a(\mathcal{O}_p^*) \cup \{\sim \psi_5, \sim \psi_{20}\}$.

Table 2 Reasoning results over buggyPolicy ontology

Axiom: ϕ/$\sim\phi$	$(\models_4, \mapsto)$	$(\models_4, \sqsubset)$	$(\models_4, \rightarrow)$	$\models_Q$	$\models_c$	$\models_b$	$\models_a$
$\psi_4/\sim\psi_4$	yes/no	yes/no	yes/yes	yes/no	no/no	yes/no	yes/no
$\psi_6/\sim\psi_6$	yes/no	yes/yes	yes/yes	yes/no	no/no	yes/no	no/yes
$\psi_{11}/\sim\psi_{11}$	yes/no	yes/yes	yes/yes	yes/no	no/no	yes/no	yes/no
$\psi_{20}/\sim\psi_{20}$	yes/no	yes/no	yes/yes	yes/no	no /no	yes/yes	no/no
$\psi_{21}/\sim\psi_{21}$	yes/no	yes/no	yes/yes	yes/no	yes/no	yes/no	yes/no
$\beta_1/\sim\beta_1$	no/no	yes/yes	yes/yes	yes/yes	no/no	yes/yes	yes/no
$\beta_2/\sim\beta_2$	no/ no	yes/no	yes/yes	yes/no	yes/no	yes/yes	yes/no
$\beta_3/\sim\beta_3$	yes/no	yes/no	yes/yes	yes/no	yes/no	yes/no	yes/no
$\beta_4/\sim\beta_4$	yes/no	yes/no	yes/yes	yes/no	yes/no	yes/yes	yes/no

As a result, *GeneralReliabilityUserPolicy* is satisfiable and these repaired ontologies are coherent and consistent.

Experiment evaluations Based on the experiment results in Tables 1 and 2, we have experiment conclusions:

- $\models_a$ brings reasonable and meaningful answers since $\models_a$ can evaluate those axioms which are possibly treated as contradictory in depth. In the badFood ontology, $\mathcal{O}_b^* \models_a \sim \alpha_1$ but $\mathcal{O}_b^* \not\models_a \alpha_1$ while the answer to α_1 is the same as that to $\sim \alpha_1$ in other entailments. In the buggyPolicy ontology, $\mathcal{O}_p^* \models_a \beta_1$ but $\mathcal{O}_p^* \not\models_a \sim \beta_1$ while the answer to β_1 is the same as that of $\sim \beta_1$ in other entailments.
- The inference power of $\models_a$ is between $\models_c$ and $\models_b$. Here we say $\models_x$ has *stronger inference power* than $\models_y$ if $\mathcal{O} \models_y \phi$ implies $\mathcal{O} \models_x \phi$ for any $\mathcal{O}$ and ϕ. $\models_b$ is strictly stronger than $\models_a$ and $\models_a$ is strictly stronger than $\models_c$. In the badFood ontology, $\mathcal{O}_b^* \not\models_a \varphi_{27}$ while $\mathcal{O}_b^* \models_b \varphi_{27}$. Moreover, $\mathcal{O}_a^* \models_a \alpha_2$ while $\mathcal{O}_b^* \not\models_c \alpha_2$. In the buggyPolicy ontology, $\mathcal{O}_p^* \not\models_a \psi_{20}$ while $\mathcal{O}_p^* \models_b \psi_{20}$. Moreover, $\mathcal{O}_p^* \models_a \psi_4$ while $\mathcal{O}_p^* \not\models_c \psi_4$.
- $\models_a$ can be used to repair ontologies. $\models_a$ can reject some axioms of an ontology, which cause inconsistency or incoherency. However both $\models_4$ and $\models_Q$ protect all axioms of $\mathcal{O}_b^*$ (i.e., all axioms can be inferred under their semantics). As a result, the argumentative entailment can be used to repair ontologies. In the badFood ontology, $\mathcal{O}_b^* \not\models_a \varphi_{27}$ and $\mathcal{O}_b^* \not\models_a \sim \varphi_{27}$. In the buggyPolicy ontology, $\mathcal{O}_p^* \not\models_a \psi_6$ and $\mathcal{O}_b^* \models_a \sim \psi_6$. Besides, $\mathcal{O}_p^* \not\models_a \psi_{20}$ and $\mathcal{O}_b^* \not\models_a \sim \psi_{20}$.

7 Related works

The issues of DL ontology reasoning and management are important in the Semantic Web. Recently, some approaches to reasoning with inconsistent DL ontologies by using argumentation are reported in several research proposals (Gómez et al. 2008, 2010; Black et al. 2009). Compared with paraconsistent approaches which employ some functions to determine which consistent subsets of an inconsistent ontology should be considered in the reasoning process (Schlobach and Cornet 2003; Huang et al. 2005), the set of warranted arguments is considered as the valid consequence in reasoning by using argumentation (see Gómez et al. 2008). Compared with multi-valued semantics of ontologies (Ma et al. 2007; Odintsov and Wansing 2008; Zhang et al. 2009, 2010; Zhang and Lin 2012), reasoning by using argumentation satisfies some good properties such as nonmonotonicity and consistency-preserving.

In Gómez et al. (2008, 2010), the Dung's argumentation framework is employed to reason with inconsistent DL ontologies. This argumentative approach starts a transformation, which is presented by Grosof et al. (2003) to translate DL axioms into rules in description logic programs (DLP) and then introduce defeasible semantics for DLP to *defeasible logic programs* (DeLP). Thus the problem about querying over DL ontologies is reduced to the problem about querying the corresponding ontologies in DeLP. The inconsistency-tolerant reasoning could be realized via the defeasible semantics of logic programs by using the dialogue mechanism of argumentation. There are at least two differences between this approach (DeLP) and our approach.

- The first difference is in handling expressive DLs. DeLP can not handle general DLs such as $\mathcal{ALC}$ since only axioms in Horn-clause logics (the intersection between DL and Horn logic programs) can be translated into rules in DLP (see Grosof et al. 2003; Gómez et al. 2008, 2010). For instance, DeLP does not support disjunctions in the head of rules (Gómez et al. 2008, 2010). Moreover, some DL axioms can not be translated into rules in DLP such as $A \sqsubseteq \exists R.B$ and $\forall R.A \sqsubseteq B$ (see Grosof et al. 2003). Because our approach directly handles DL ontologies without those restriction of the transform, we can handle general DLs such as $\mathcal{ALC}$ and we argue that our scenario would be technically feasible for more expressive DLs. Additionally, our approach handles incoherent ontologies.
- The second difference is in handling mechanism. DeLP is based on Dung's argumentation framework (a graph-based structure) (Dung 1995b) while our framework is built on BH's framework (a tree-based structure) (Besnard and Hunter 2001). Semantically, each axiom in DeLP has one of the two values ("true" and "false") while our semantics is multi-valued (see Theorem 5). Additionally, our semantics does not bring possible conflicts between open world semantics in DLs and closed world semantics (LP).

In Black et al. (2009), the BH's argumentation framework is applied to reason with multiple ontologies. Consider two DL ontologies which (possibly) contradicts to each other, and they are taken as two agents. Queries over the two ontologies will be answered by using negotiation of the two agents. Though both this approach and our approach are employing BH's argumentation framework, there are at least two differences between this approach and our approach.

- The first difference is in handling objects. Because this approach is proposed to querying over multiple ontologies where each ontology should be coherent and consistent, it is not suitable for paraconsistent reasoning with a single incoherent or inconsistent ontology.
- The second difference is in reasoning states. This semantics implicitly presented is based on three values since the answer to a query has three states, namely, *true*, *false* and *unknown* while our semantics is based on four values.

There are some syntax-based and model-based approaches to repairing DL ontologies with maintaining coherency or consistency. On the one hand, those syntax-based methods based on consistent subsets (Schlobach 2005; Meyer et al. 2006; Kalyanpur et al. 2006b; Du and Shen 2008) are difficult to keep semantics closer to classical semantics. On the other hand, those model-based approaches (Qi and Du 2009; Ji et al. 2009; Wang et al. 2010) are hard to maintaing the syntactic structure so that their repaired ontologies might be in short of readability. Moreover, those revising approaches (Qi and Du 2009; Wang et al. 2010) are not suitable for repairing single ontology but two ontologies. Compared with those existing approaches, we propose a justifiable mechanism to repair ontologies. Within it, we give consideration to both semantic closeness and syntactical readability. Moreover, our proposal could handle incoherency and inconsistency of general DL ontologies in a unified way.

8 Conclusion and the future work

In this paper, we have presented an argumentation framework to handle inconsistent/incoherent DL ontologies. Within this framework, we can reason with tolerating inconsistent knowledge in a justifiable way. Moreover, we can also maintain the coherency and consistency of ontologies again by using some repairing operators developed under this framework: one normal operator is developed to compute the repaired ontologies with keeping justifiability; and two approximative operators are presented to enrich the normal operator by adding much more information. The novel approach not only would provide an alternative scenario for DL ontology management and reasoning with giving consideration to both semantics and syntax but also might enlighten some potential proposals in merging DL ontologies which is still an open problem. The basic problem of implementing our proposed reasoning is searching and generating arguments. A naive approach to generating arguments of an axiom in a DL ontology is applying existing DL reasoners Sirin et al. (2007) to checking all subsets of this ontology. However, it would be low-efficient because the support of an argument is generally computed in exponential times. Finding an efficient approach to generating arguments in DL ontologies and then finally implementing a system to serve the proposed reasoning will be considered as our future work.

Acknowledgements We would like to thank the anonymous referees for their critical comments which helped us to improve the paper. We thank Kedian Mu, Yue Ma, Zhihu Zhang and Dai Xu etc for helpful comments and discussions. Xiaowang Zhang is funded by the project of Research Foundation Flanders under grant G.0489.10N and Zuoquan Lin is funded by the Ph.D. Programs Foundation of Ministry of Education of China.

References

Arieli, O., Avron, A., Zamansky, A. (2011). What is an ideal logic for reasoning with inconsistency? In *Proc. of IJCAI'11, USA* (pp. 706–711). Menlo Park: AAAI Press.

Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (Eds.) (2003). *The description logic handbook: theory, implementation, and applications*. Cambridge: Cambridge University Press.

Berners-Lee, T., Hendler, J., Lassila, O. (2001). *The semantic web*. New York: Scientific American Magazine.

Bertossi, L.E. , Hunter, A., Schaub, T. (Eds.) (2005). *Inconsistency tolerance. LNCS 3300*. New York: Springer.

Besnard, P., & Hunter, A. (2001). A logic-based theory of deductive arguments. *Artificial Intelligence, 128*(1–2), 203–235.

Bienvenu, M. (2008). Prime implicate normal form for ALC concepts. In *Proc. of AAAI'08, USA* (pp. 412–417). Menlo Park: AAAI Press.

Black, E., Hunter, A., Pan, J.Z. (2009). An argument-based approach to using multiple ontologies. In *Proc. of SUM'09, USA, LNAI 5785* (pp. 68–79). New York: Springer.

Calvanese, D. (1996). Finite model reasoning in description logics. In *Proc. of KR'96, USA* (pp. 292–303). San Mateo: Morgan Kaufmann.

Davies, J., Grobelnik, M., Mladenic, D. (Eds.) (2009). *Semantic knowledge management. Integrating ontology management, knowledge discovery, and human language technologies*. New York: Springer.

Du, J., & Shen, Y. (2008). Computing minimum cost diagnoses to repair populated DL-based ontologies. In *Proc. of WWW'08, China* (pp. 565–574). New York: ACM.

Dung, P.M. (1995a). An argumentation-theoretic foundations for logic programming. *Journal of Logic Programming, 22*(2), 151–171.

Dung, P.M. (1995b). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence, 77*(2), 321–358.

Flouris, G., Huang, Z., Pan, J.Z., Plexousakis, D., Wache, H. (2006). Inconsistencies, negations and changes in ontologies. In *Proc. of AAAI'06, USA*. Menlo Park: AAAI Press.

Gómez, S.A. , Chesñevar, C.I. , Simari, G.R. (2008). An argumentative approach to reasoning with inconsistent ontologies. In *Proc. of KROW'08, Australia, CRPIT 90* (pp. 11–20). Nottingham: ACS.

Gómez, S.A., Chesñevar, C.I., Simari, G.R. (2010). Reasoning with inconsistent ontologies through argumentation. *Applications of Artificial Intelligence, 24*(1–2), 102–148.

Grosof, B.N., Horrocks, I., Volz, R., Decker, S. (2003). Description logic programs: combining logic programs with description logic. In *Proc. of WWW'03, Hungary* (pp. 48–57). New York: ACM.

Horrocks, I. (2008). Ontologies and the semantic web. *Communications of the ACM, 51*(12), 58–67.

Horrocks, I., & Patel-Schneider, P.F. (2003). Reducing OWL entailment to description logic satisfiability. In *Proc. of ISWC'03, USA, LNCS 2870* (pp. 17–29). New York: Springer.

Huang, Z., van Harmelen, F., ten Teije, A. (2005). Reasoning with inconsistent ontologies. In *Proc. of IJCAI'05, UK* (pp. 454–459). Professional Book Center.

Ji, Q., Haase, P., Qi, G., Hitzler, P., Stadtmüller, S. (2009). RaDON—repair and diagnosis in ontology networks. In *Proc. of ESWC'09, Greece, LNCS 5554* (pp. 863–867). New York: Springer.

Kalyanpur, A., Parsia, B., Sirin, E., Grau, C.B. (2006). Debugging unsatisfiable classes in OWL ontologies. *Journal of Web Semantics, 3*(4), 268–293.

Kalyanpur, A., Parsia, B., Sirin, E., Grau, C.B. (2006). Repairing unsatisfiable concepts in OWL ontologies. In *Proc. of ESWC'06, Montenegro, LNCS 4011* (pp. 170–184). New York: Springer.

Konev, B., Walther, D., Wolter, F. (2009). Forgetting and uniform interpolation in large-scale description logic terminologies. In *Proc. of IJCAI'09, USA* (pp. 830–835).

Ma, Y., Hitzler, P., Lin, Z. (2007). Algorithms for paraconsistent reasoning with OWL. In *Proc. of ESWC'07, Austria, LNCS 4519* (pp. 399–413). New York: Springer.

Meyer, T., Lee, K., Booth, R. (2005). Knowledge integration for description logics. In *Proc.of AAAI'05, USA* (pp. 645–650). Cambridge: AAAI Press/The MIT Press.

Meyer, T., Lee, K., Booth, R., Pan, J. (2006). Finding maximally satisfiable terminologies for the description logic ALC. In *Proc. of AAAI'06, USA*. Menlo Park: AAAI Press.

Odintsov, S.P., & Wansing, H. (2008). Inconsistency-tolerant description logic. part II: a tableau algorithm for CACL^c. *Journal of Applied Logic, 6*(3), 343–360.

Patel-Schneider, P.F. (1989). A four-valued semantics for terminological logics. *Artificial Intelligence, 38*(3), 319–351.

Qi, G., & Du, J. (2009). Model-based revision operators for terminologies in description logics. In *Proc. of IJCAI'09, USA* (pp. 891–897). Professional Book Center.

Schlobach, S. (2005). Diagnosing terminologies. In *Proc. of AAAI'05, USA*. Cambridge: AAAI Press/The MIT Press.

Schlobach, S., & Cornet, R. (2003). Non-standard reasoning services for the debugging of description logic terminologies. In *Proc. of IJCAI'03, Mexico* (pp. 355–362). San Mateo: Morgan Kaufmann.

Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics, 5*(2), 51–53.

Wang, Z., Wang, K., Topor, R.W. (2010). A new approach to knowledge base revision in DL-Lite. In *Proc. of AAAI'10, USA*. Menlo Park: AAAI Press.

Zhang, X., & Lin, Z. (2012). Quasi-classical description logic. *Journal of Multiple-Valued Logic and Soft Computing, 18*(3–4), 291–327.

Zhang, X., Lin, Z., Wang, K. (2010). Towards a paradoxical description logic for the Semantic Web. In *Proc. of FoIKS'10, Bulgaria, LNCS 5956* (pp. 306–325). New York: Springer.

Zhang, X., Xiao, G., Lin, Z. (2009). A tableau algorithm for handling inconsistency in OWL. In *Proc. of ESWC'09, Greece, LNCS 5554* (pp. 399–413). New York: Springer.