Computing and Informatics, Vol. 32, 2013, 1001–1025, V 2013-Oct-1

AN APPROACH TO GENERATING ARGUMENTS OVER DL-LITE ONTOLOGIES

Xiaowang Zhang

Databases and Theoretical Computer Science Group
Hasselt University and Transnational University of Limburg
Diepenbeek 3590, Belgium
e-mail: xiaowang.zhang@uhasselt.be

Zuoquan Lin

School of Mathematical Sciences, Peking University
Beijing 100871, P. R. China
e-mail: lz@math.pku.edu.cn

Communicated by Jacek Kitowski

Abstract. Argumentation frameworks for ontology reasoning and management have received extensive interests in the field of artificial intelligence in recent years. As one of the most popular argumentation frameworks, Besnard and Hunter's framework is built on arguments in the form of $\langle \Phi, \phi \rangle$ where Φ is consistent and minimal for entailing ϕ. However, the problem of generating arguments over ontologies is still open. This paper presents an approach to generating arguments over DL-Lite ontologies by searching support paths in focal graphs. Moreover, theoretical results and examples are provided to ensure the correctness of this approach. Finally, we show that approach has the same complexity as propositional revision.

Keywords: DL-Lite, ontology, argument, focal graph, support path

1 INTRODUCTION

Description logic Lite (DL-Lite) [5], which is a family of lightweight description logics (DLs) [8], is the logical foundation of OWL 2.0 QL – one of the three profiles

of OWL 2.0 for Web ontology language recommended by W3C [18]. The DL-Lite family is specially tailored for efficient reasoning and, at the same time, relatively high expressing power. DL-Lite provides constructors which are necessary to express many conceptual models such as Entity-Relationship Model (ERM) and Unified Modeling Language (UML) class diagrams.

Inconsistency is not rare in ontology applications [7, 24] and may be caused by several reasons, such as errors in modeling, migration from other formalisms, ontology merging, and ontology evolution [20, 9, 10, 31]. Unfortunately, based on standard inference, the inference of DL-Lite encounters the problem when inconsistency occurs, which is referred to as the triviality problem. That is, any conclusions, that are possibly irrelevant or even contradicting, will be entailed from an inconsistent DL-Lite ontology under the standard inference. As one kind of fundamental methods of handling inconsistency in DLs, *paraconsistent approach* is applying a non-standard inference to obtain meaningful answers from inconsistent ontologies [17, 30, 29, 26]. In recent days, some paraconsistent approaches based on logic-based argumentation are presented to deal with inconsistent ontologies [4, 14, 27, 6, 15] by applying some dialogue mechanisms (e.g., Dung's framework [13] and Besnard and Hunter's (BH's) framework [12]) to assess inconsistent information and information causing inconsistency. A common assumption for logic-based argumentation is that an argument contains two parts: support and consequent [12]. Formally, an argument $\mathbf{A}$ for some axiom ϕ in some ontology $\mathcal{O}$ is a pair $\langle \Phi, \phi \rangle$, where both Φ is a minimal consistent subset of $\mathcal{O}$ and Φ is an implicant of ϕ (i.e., $\Phi \models \phi$). The problem about generating arguments over ontologies is important since it is the first step to construct BH's framework. However, this problem is still open in DLs.

Recently an approach based on connection graphs presented by Efstathiou and Hunter [21, 23] (for short, the EH's approach) has been developed to search and generate arguments for propositional axioms. However, it is not straightforward to adapt this approach in DL-Lite, since arguments for DL-Lite complex axioms, namely, concept inclusions in the form of $C \sqsubseteq D$, concept assertions in the form of $C(a)$ and role assertions in the form of $R(a, b)$ where C, D are concepts, R a role and a, b are individuals, are not directly captured within the connection graphs. For instance, it is infeasible to search arguments whose either support or consequent contains some role assertions within a connection graph since each node of connection graph describes an axiom with respect to a single individual. Additionally, a DL-Lite ontology often has infinite number of models and some models might also be infinite.

To overcome these difficulties, in this paper we develop an edge-labeled connection graph to search and generate arguments over DL-Lite ontologies. In our previous proceedings [25], we present an approach to obtaining a searching scope of all arguments for some axiom in a given ontology. Compared with [25], this paper refines and extends it by adding an approach to generating all arguments. The main innovations and contributions of this paper can be summarized as follows. Firstly, we present an edge-labelled connect graph called *attack graph* to characterize a DL-Lite ontology. Within an attack graph, we show that the resolutive relation between

two DL axioms can be equivalently captured by the edge that connects them (where axioms are taken as vertices). Secondly, we define a closed attack graph (called *closed graph*) to avoid those redundant vertices and, based on the labels of edges, we then introduce a *focal graph* within a closed graph to remove irrelevant edges to cut down the search space for arguments. We prove that all arguments for an axiom are indeed captured by its focal graph. Finally, we define *support paths* to generate arguments by further selecting appropriate paths in a focal graph. Each support path can be proved to correspond to an argument. Moreover, our approach does not bring a higher complexity than that of the EH's approach to generating arguments for propositional axioms. Besides, we have provided theoretical results and examples to ensure the correctness of this approach.

This paper is structured as follows. The following section briefly reviews the DL-Lite family and arguments. Section 3 presents attack graphs over ontologies. Section 4 introduces an approach to searching and generating arguments and Section 5 applies our proposal approach to discuss a practical example. The last Section 6 concludes this paper and discusses our future work.

2 PRELIMINARIES

In this section, we briefly review the DL-Lite family and arguments over ontologies. For more comprehensive background knowledge of DLs, we refer the reader to some basic references such as the DL-Lite family [1, 8] and arguments [27, 6].

2.1 The DL-Lite Family

Description logics are formal knowledge representation languages whose expressive powers are between propositional logic and first-order logic [8]. DLs can represent the domain of interest in terms of concepts, denoting sets of objects, and roles, denoting binary relations between (instances of) concepts. Complex concept and role expressions are constructed starting from a set of atomic concepts and roles by applying suitable constructors. Different DLs allow for different constructors. Properties of concepts and roles can be specified through inclusion assertions, stating that every instance of a concept (or role) is also an instance of another concept (or role).

As an important family of DLs, DL-Lite is specifically tailored to capture basic ontology languages, while keeping all reasoning tasks tractable. In this paper, we consider the core language for the whole family of DL-Lite: DL-Lite$_{core}$.

Let N_C, N_R and N_I be pairwise disjoint and countably infinite sets of *concept names*, *role names*, and *individual names*, respectively. Concepts and roles are inductively constructed according to the following rules:

$$\begin{aligned} \text{role:} \quad & R \rightarrow P \mid P^{-}; \\ \text{basic concept:} \quad & B \rightarrow A \mid \exists R; \\ \text{general concept:} \quad & C \rightarrow B \mid \neg B; \end{aligned}$$

where $A \in N_C$, $P \in N_R$, and P^- the *inverse* of the atomic role P. B denotes a *basic concept*, that is, a concept that can be either an atomic concept or a concept of the form $\exists R$. R denotes a basic role, that is, a role that is either an atomic role or the inverse of an atomic role. Note that $\exists R$ is the standard DL constructor of unqualified existential quantification on basic roles. We write $R^- = P^-$ if $R = P$, and $R^- = P$, if $R = P^-$. We also write $\neg C = \neg A$ if $C = A$ (or $\neg E = \neg R$ if $E = R$), and $\neg C = A$ if $C = \neg A$ (or $\neg E = R$, if $E = \neg R$).

In this paper, we use the letters A (with subscripts) for a concept name, B for a basic concept, the letters P for a role name, the letters C, D for general concepts, R for a basic role, E for a general role, a, b for (named) individuals, x, y for individual variables and t, s for (named) individuals or individual variables. Let $\top$ and $\bot$ denote the *universal concept* and the *bottom concept*, respectively.

A DL *ontology* $\mathcal{O}$ contains two parts, namely *ABox* and *TBox*, where ABox is used to represent extensional information and TBox is used to represent intensional knowledge. A TBox is a finite set of *concept inclusions* in the form of $B \sqsubseteq C$ and an ABox is a finite set of *assertions*, namely, *concept assertions* in the form of $C(a)$ and *role assertions* in the form of $P(a, b)$. For instance, $Professor \sqsubseteq \exists teachesTo$ is a concept inclusion, $Professor(russell)$ is a concept assertion and $teachesTo(russell, wittgenstein)$ is a role assertion. Concept inclusions, general concept assertions, role assertions and inverse role assertions are called *axioms*. An axiom is called *positive* if it contains no $\neg$, and *negative* otherwise. For instance, $Professor(russell)$ is a positive axiom while $\neg\exists hasChildren(dink)$ is a negative axiom. A *sub-ontology* $\mathcal{O}'$ of $\mathcal{O}$ is an ontology whose axioms are in $\mathcal{O}$, denoted by $\mathcal{O}' \subseteq \mathcal{O}$. In this sense, we also say $\mathcal{O}$ is a *sup-ontology* of $\mathcal{O}'$. If, in addition, $\mathcal{O}' \neq \mathcal{O}$, we write $\mathcal{O}' \subset \mathcal{O}$. We use $\mathcal{O} \cup \mathcal{S}$ and $\mathcal{O} \setminus \mathcal{S}$ to denote the *union* and *difference* of $\mathcal{O}$ and $\mathcal{S}$, respectively.

The semantics of a DL is given in terms of interpretations, where an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty interpretation domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$ that assigns to each A a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, and to each P a binary relation $P^{\mathcal{I}}$ over $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ and satisfies the following conditions:

$$\begin{aligned} P^{-\mathcal{I}} &= \{(x, y) \mid (y, x) \in P^{\mathcal{I}}\}; \\ (\neg B)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus B^{\mathcal{I}}; \\ \exists R^{\mathcal{I}} &= \{x \mid \exists y.\, (x, y) \in R^{\mathcal{I}}\}. \end{aligned}$$

An interpretation $\mathcal{I}$ is a *model* of an/a inclusion axiom (concept assertion, role assertion) $B \sqsubseteq C$ ($C(a)$, $P(a, b)$) if $B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ ($a^{\mathcal{I}} \in C^{\mathcal{I}}$, $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$), denoted by $\mathcal{I} \models B \sqsubseteq C$ ($\mathcal{I} \models C(a)$, $\mathcal{I} \models R(a, b)$), respectively. We say $C^{\mathcal{I}}$ an *extension* of C with respect to $\mathcal{I}$. Similarly, $\mathcal{I}$ is a *model* of $C_1 \sqsubseteq C_2$ ($C(a)$) if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ ($a^{\mathcal{I}} \in C^{\mathcal{I}}$).

An interpretation $\mathcal{I}$ is a *model* of TBox $\mathcal{T}$ (ABox $\mathcal{A}$) if $\mathcal{I}$ is a model of all axioms in $\mathcal{T}$ ($\mathcal{A}$), denoted by $\mathcal{I} \models \mathcal{T}$ ($\mathcal{I} \models \mathcal{A}$), respectively. Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology. An interpretation $\mathcal{I}$ is a *model* of $\mathcal{O}$ if $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$. Let $Mod(\mathcal{O})$ be the set of all models of $\mathcal{O}$. $\mathcal{O}$ is *consistent* if $Mod(\mathcal{O}) \neq \emptyset$, and *inconsistent* otherwise. Let ϕ be an axiom. $\mathcal{O}$ *entails* ϕ, denoted by $\mathcal{O} \models \phi$, if $Mod(\mathcal{O}) \subseteq Mod(\{\phi\})$, that

is, every model of $\mathcal{O}$ is a model of ϕ. ϕ is a *contradiction* if $\{\phi\}$ is inconsistent. A contradiction has the form of $\bot(t)$ or $\top \sqsubseteq \bot$. So conversely, a *tautology* has the form of $\top(t)$, $\bot \sqsubseteq C$, $B \sqsubseteq \top$, $B \sqsubseteq B$ or $\bot \sqsubseteq \top$.

In DL-Lite, there are some basic reasoning tasks. The *consistency* problem is deciding whether an ontology is consistent or not. Two checking problems, namely, *instance checking* ($\mathcal{O} \models C(a)$) and *subsumption checking* ($\mathcal{O} \models B \sqsubseteq C$), can be reduced to the consistency problem by the following lemma [8].

Lemma 1 (Reduction)**.** Let $\mathcal{O}$ be an ontology. We have

- $\mathcal{O} \models B \sqsubseteq C$ if and only if $\mathcal{O} \cup \{B(\tau), \neg C(\tau)\}$ is inconsistent where τ is a new individual not occurring in $\mathcal{O}$;
- $\mathcal{O} \models C(a)$ if and only if $\mathcal{O} \cup \{\neg C(a)\}$ is inconsistent.

At the end of this section, to specify the semantics, we enforce the *unique name assumption* (UNA), that is, for any two individuals a, b and any interpretation $\mathcal{I}$, $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. For instance, based on UNA, *russell* and *wittgenstein* are always treated as two different persons (individuals).

2.2 Arguments

Let $\mathcal{O}$ be an ontology and ϕ an axiom. An *argument* for ϕ with respect to $\mathcal{O}$ is a pair $\langle \Phi, \phi \rangle$ such that the following three axioms hold true:

- *entailment*: $\Phi \models \phi$;
- *consistency*: Φ is consistent;
- *minimality*: for any $\Phi' \subset \Phi$, $\Phi' \not\models \phi$.

If $\mathbf{A} = \langle \Phi, \phi \rangle$ is an argument, then we call Φ the *support* of $\mathbf{A}$ and ϕ the *consequent* of $\mathbf{A}$. $Sup(\mathbf{A}) = \Phi$ and $Con(\mathbf{A}) = \phi$. Let $Arg(\mathcal{O}, \phi)$ denote the set of all arguments for ϕ with respect to $\mathcal{O}$. We will simply say "arguments" whenever it is clear from the context in which ontology we work.

Intuitively, the first condition states that the support can entail the consequent; the second shows that the support of an argument is a consistent sub-ontology; and the last ensures that the support is minimal among all sub-ontologies which can entail the consequent. In other words, the support of an argument for some axiom is a minimal consistent sub-ontology which can entail this axiom (consequent).

Example 1 (Pet ontology)**.** Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology where $\mathcal{T} = \{Dog \sqsubseteq Pet, Cat \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner, \exists hasOwner^- \sqsubseteq Person, Cat \sqsubseteq \neg Dog\}$ and $\mathcal{A} = \{\neg Person(p), Dog(d), hasOwner(d, p), Cat(c), \neg\exists hasOwner(c)\}$ where *Dog*, *Pet*, *Cat*, *Person* are concepts, *hasOwner* a role and c, d, p individuals. The intended meaning of $\mathcal{O}$ is as follows: a dog is a pet; a cat is a pet; a pet has a owner who is a person; a cat is not a dog; d is a dog; d has a owner p; p is not a person; c is is a cat; and c has not an owner.

It is hard to show that $\mathcal{O}$ is inconsistent. $hasOwner(d, p)$ and $\exists hasOwner^- \sqsubseteq Person$ imply $Person(p)$ while $\neg Person(p)$ is an axiom. Moreover, $Cat(c)$, $Cat \sqsubseteq Pet$, $Pet \sqsubseteq \exists hasOwner$ imply $\exists hasOwner(c)$ while $\neg\exists hasOwner(c)$ is an axiom.

Five arguments $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$, $\mathbf{A}_4$ and $\mathbf{A}_5$ are constructed as follows:

1. $\mathbf{A}_1 = \langle \Phi_1, \phi_1 \rangle$, where

$$\begin{array}{rcl} \Phi_1 & = & (\emptyset, \{hasOwner(d, p)\}); \\ \phi_1 & = & \exists hasOwner(d). \end{array}$$

   Intuitively, if d has an owner p, then d has an owner.

2. $\mathbf{A}_2 = \langle \Phi_2, \phi_2 \rangle$, where

$$\begin{array}{rcl} \Phi_2 & = & (\{Dog \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner\}, \{Dog(d)\}); \\ \phi_2 & = & \exists hasOwner(d). \end{array}$$

   Intuitively, if d is a dog, each dog is a pet, and each pet has an owner, then d has an owner.

3. $\mathbf{A}_3 = \langle \Phi_3, \phi_3 \rangle$, where

$$\begin{array}{rcl} \Phi_3 & = & (\{\exists hasOwner^- \sqsubseteq Person\}, \{hasOwner(d, p)\}); \\ \phi_3 & = & Person(p). \end{array}$$

   Intuitively, if d has an owner p and each owner is a person then p is also a person.

4. $\mathbf{A}_4 = \langle \Phi_4, \phi_4 \rangle$, where

$$\begin{array}{rcl} \Phi_4 & = & (\{Cat \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner\}, \{Cat(c))\}); \\ \phi_4 & = & \exists hasOwner(c). \end{array}$$

   Intuitively, if c is a cat, each cat is a pet and each pet has an owner then c has an owner.

5. $\mathbf{A}_5 = \langle \Phi_5, \phi_5 \rangle$, where

$$\begin{array}{rcl} \Phi_5 & = & (\{Dog \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner\}, \emptyset); \\ \phi_5 & = & Dog \sqsubseteq \exists hasOwner. \end{array}$$

   Intuitively, if each dog is a pet and each pet has an owner then each dog has an owner.

3 ATTACK GRAPHS OVER DL-LITE ONTOLOGIES

The BH's framework [12] provides a tree-based dialogue mechanism, which can ensure the termination of dialogue mechanisms, to deal with conflicts occurring in knowledge.

Formally, the BH's framework $\mathcal{F}$ for some axiom ϕ over some ontology $\mathcal{O}$ is a pair $\mathcal{P}, \mathcal{C}$ where $\mathcal{P}$ and $\mathcal{C}$ are sets of argument trees for and against ϕ in $\mathcal{O}$ [12, 27]. Arguments are basic in constructing argument trees since each vertex in an argument tree is an argument.

To search and generate arguments over ontologies, we need to discuss three kinds of DL axioms, namely, concept inclusions "$B \sqsubseteq C$", role assertions "$P(a, b)$", and concept assertions "$B(a)$".

Because for any role assertion $P(a, b)$ (or $P^-(b, a)$), the argument for it is $\langle \{P(a, b)\}, P(a, b)\rangle$ (or $\langle \{P(a, b)\}, P^-(b, a)\rangle$), it is trivial to search arguments for role assertions in DL-Lite$_{core}$. In this paper, we mainly consider arguments for concept assertions and concept inclusions. The problem of searching all arguments for an axiom ϕ with respect to an ontology $\mathcal{O}$ is indeed computing their supports which satisfy three properties: entailment, consistency, and minimality.

To address this issue, we will compute supports in four steps by gradually narrowing our searching scope and finally generating arguments within this scope:

- introducing an edge-labeled connection graph called *attack graph* via *resolution rule* where each vertex is an axiom and there exists an edge between two axioms if and only if resolution rules can be applied over them;
- defining a kind of subgraphs called *closed graphs* of a connection graph where each vertex has always an edge with other vertex in order to characterize entailment relationship (discussed in Section 4.1);
- presenting a further kind of subgraphs called *focal graphs* of a closed graph by removing redundant vertexes to characterize minimality and consistency. Then, we will show that the support of each argument belongs to a focal graph (discussed in Section 4.1);
- selecting some paths called *support paths* in a focal graph by removing redundant vertexes and edges. Finally, we will show that an argument can be exactly generated for each support path (discussed in Section 4.2).

In this section, we will introduce attack graphs of DL-Lite ontologies where each axiom is taken as a node. To define edges between nodes, we will introduce resolution rules which are rules of inference leading to a refutation theorem-proving technique for axioms [3]. In DL-Lite$_{core}$, there are four kinds of resolution rules as follows:

- rule-1:
$$\frac{C(x), \neg C(x)}{\bot(x)} \quad \text{or} \quad \frac{C(x), \neg C(a)}{\bot(a)};$$
- rule-2:
$$\frac{R(a, b), \neg \exists R(t)}{\bot(a)} \quad \text{or} \quad \frac{R(b, a), \neg \exists R^-(t)}{\bot(a)};$$
- rule-3:
$$\frac{B(t), B \sqsubseteq C}{C(t)} \quad \text{or} \quad \frac{\neg C(t), B \sqsubseteq C}{\neg B(t)};$$

- rule-4:
$$\frac{B_1 \sqsubseteq B_2, B_2 \sqsubseteq C}{B_1 \sqsubseteq C} \quad \text{or} \quad \frac{B_1 \sqsubseteq \neg C, B_2 \sqsubseteq C}{B_1 \sqsubseteq \neg B_2}.$$

Resolution rules can be used to determine the inconsistency of an ontology in the following Lemma [3].

Lemma 2 (Resolution)**.** Let $\mathcal{O}$ be an ontology. $\mathcal{O}$ is inconsistent if and only if a contradiction (in forms of $\bot(t)$, $\top \sqsubseteq \bot$) is obtained by exhaustedly applying resolution rules.

Lemma 2 states that an inconsistent ontology can finally bring a contradiction via resolution rules.

By combining Lemma 1 and Lemma 2, we conclude the following results: for every ontology $\mathcal{O}$,

1. $\mathcal{O} \models B \sqsubseteq C$ if and only if a contradiction in forms of $\bot(x)$, $\bot(\tau)$ is obtained from $\mathcal{O} \cup \{B(\tau), \neg C(\tau)\}$ by exhaustedly applying resolution rules;
2. $\mathcal{O} \models C(a)$ if and only if a contradiction in forms of $\bot(x)$, $\bot(a)$ is obtained from $\mathcal{O} \cup \{\neg C(a)\}$ by exhaustedly applying resolution rules.

If two axioms ϕ, ψ meet the precondition of resolution rules, then we simply say that the pair (ϕ, ψ) is *resolutive*. In other words, two axioms of a resolutive pair can invoke one resolution rule. In this sense, if the pair (ϕ, ψ) is resolutive then there exists the *resolutive relation* between ϕ and ψ.

Next, we investigate some common features of all resolutive pairs.

For instance, let $\mathcal{T} = \{\top \sqsubseteq B, \top \sqsubseteq \neg B\}$ be a TBox. Obviously, $\mathcal{T}$ is inconsistent, that is, there exists no model of $\mathcal{T}$. In general, inconsistency is caused by some conflicts.

Formally, a *conflict* is a set of $\{B(t), \neg B(t)\}$. If the consistency of $\mathcal{T}$ is caused by some conflicts $\{B(t), \neg B(t)\}$, then we say $\mathcal{T}$ *contains* $\{B(x), \neg B(x)\}$. For instance, $\{\top \sqsubseteq B, \top \sqsubseteq \neg B\}$ contains conflicts in form of $\{B(t), \neg B(t)\}$.

Indeed, a DL-Lite$_{core}$ ontology contains conflicts in two forms as follows:

$$\textbf{conflict-1: } \{A(t), \neg A(t)\}; \qquad \textbf{conflict-2: } \{\exists R(t), \neg \exists R(t)\}.$$

Specially, $\{X(a), \neg X(x)\}$ and $\{X(x), \neg X(a)\}$ are still conflicts where $X \in \{A, \exists R\}$. However, $\{C(a), \neg C(b)\}$ is not a conflict. In other words, conflicts are related to individuals. For instance, $\{C(x), \neg C(x)\}$ is a conflict. We use $ID(\phi)$ to return the individual name occurring in a concept assertion ϕ. For convenience, we still denote $ID(X(x)) = x$. For instance, $ID(Dog(d)) = d$ and $ID(Cat(c)) = c$.

The precondition of an arbitrary resolution rule contains a conflict.

Proposition 1. Let ϕ and ψ be two axioms. The pair (ϕ, ψ) is resolutive if and only if $\{\phi, \psi\}$ contains a conflict.

Given an arbitrary pair (ϕ, ψ), it is not obvious to find all conflicts included in $\{\phi, \psi\}$ since there are three kinds of axioms. For instance, the pair $(P(a,b), \exists P^- \sqsubseteq C)$ is resolutive since $\{\exists P^-(b), \exists P^-(x)\}$ is a conflict.

Because a conflict is related to some basic concept assertions, we introduce two functions: $\Gamma(\cdot)$ and $\Theta(\cdot)$. The former is used to extract basic concept assertion from both role assertions and concept inclusions; and the latter is used to return the positive basic concept assertion of a conflict.

Firstly, we introduce a function *Disjunct* to beforehand transform both role assertions and concept inclusions into concept assertions.

Let ϕ be an axiom. $Disjunct(\phi)$ is defined as follows:

- $Disjunct(C(t)) = \{C(t)\}$;
- $Disjunct(P(a,b)) = \{\exists P(a), \exists P^-(b)\}$;
- $Disjunct(B \sqsubseteq C) = \{\neg B(x), C(x)\}$.

Each member of $\Gamma(\phi)$ is called a *base* of ϕ. Note that each base is a basic concept assertion. If a base is without $\neg$ then we call it a *positive base*.

$Disjunct(\phi)$ does not necessarily keep all models of $\{\phi\}$ but extracts those implied concept assertions from it. For instance, $Mod(\{P(a,b)\}) \neq Mod(\{\exists P(a), \exists P^-(b)\})$.

Besides, $Disjunct(\phi)$ possibly contains some tautologies or contradictions with $\top$ or $\bot$. For instance, $Disjunct(\top \sqsubseteq C) = \{\bot(x), C(x)\}$, $\Gamma(B \sqsubseteq \bot) = \{\neg B(x), \bot(x)\}$ and $Disjunct(B \sqsubseteq \top) = \{\neg B(x), \top(x)\}$. However, they do not support any useful information in reasoning. To simplify our discussion, we will directly remove those tautologies or contradictions in $Disjunct(\phi)$.

$\Gamma(\phi)$ is obtained from $Disjunct(\phi)$ by applying the following rules:

- if $\bot(t) \in Disjunct(\phi)$, then we set $\Gamma(\phi) = Disjunct(\phi) - \{\bot(t)\}$;
- if $\top(t) \in Disjunct(\phi)$, then we set $\Gamma(\phi) = \emptyset$.

Thus $\Gamma(\phi)$ contains neither a tautology nor a contradiction. For instance, $\Gamma(\top \sqsubseteq C) = \{C(x)\}$, $\Gamma(B \sqsubseteq \bot) = \{\neg B(x)\}$, $\Gamma(B \sqsubseteq \top) = \emptyset$.

Let $\mathcal{S}$ be a conflict. $\Theta(\mathcal{S})$ is defined as follows:

- $\Theta(\{X(a), \neg X(x)\}) = X(a)$;
- $\Theta(\{\neg X(a), X(x)\}) = X(a)$;
- $\Theta(\{X(x), \neg X(x)\}) = X(x)$.

Therefore we can obtain all positive bases from an arbitrary resolutive pair of two axioms (which contains some conflicts) by using functions Γ and Θ.

Next, we introduce a function *PreAttack* to collect all positive bases of two axioms which construct a resolutive pair.

Definition 1 (PreAttack)**.** Let ϕ and ψ be two axioms. The *preattack* of ϕ and ψ, denoted by $PreAttack(\phi, \psi)$, is a set of basic concept assertions defined as follows:

$$PreAttack(\phi, \psi) = \{\Theta(\{\alpha, \neg\alpha\}) \mid \{\alpha, \neg\alpha\} \text{ is a conflict and } \alpha \in \Gamma(\phi), \neg\alpha \in \Gamma(\psi)\};$$

where $\alpha \in \{A(t), \exists R(t), \neg A(t), \neg\exists R(t)\}$.

For instance, in Pet ontology (shown in Example 1),

1. $PreAttack(Dog(d), Dog \sqsubseteq Pet) = \{Dog(d)\}$;
2. $PreAttack(Cat \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner) = \{Pet(x)\}$;
3. $PreAttack(\neg Person(p), \exists hasOwner^{-} \sqsubseteq Person) = \{Person(p)\}$;
4. $PreAttack(hasOwner(d, p), \exists hasOwner^{-} \sqsubseteq Person) = \{\exists hasOwner^{-}(p)\}$.

PreAttack can indeed characterize all resolutive pairs of axioms.

Proposition 2. Let ϕ, ψ be two axioms. $PreAttack(\phi, \psi) \neq \emptyset$ if and only if the pair (ϕ, ψ) is resolutive.

Though *PreAttack* can characterize all resolutive pairs of axioms, some of them are redundant to meet our aim. For instance, let $\phi_1 = B_1 \sqsubseteq B_2$ and $\phi_2 = B_2 \sqsubseteq B_1$ be two concept inclusions. Thus $PreAttack(\phi_1, \phi_2) = \{B_1(x), B_2(x)\}$. After applying resolution rule **rule-4**, we have

$$\frac{B_1 \sqsubseteq B_2,\ B_2 \sqsubseteq B_1}{B_1 \sqsubseteq B_1}.$$

However, $B_1 \sqsubseteq B_1$ is a tautology. In other words, the resolution between ϕ_1 and ϕ_2 does not help in searching arguments.

We define a new function *Attack* to refine *Preattack* by removing those useless pairs.

Definition 2 (Attack)**.** Let ϕ and ψ be two axioms. The *attack* of ϕ and ψ, denoted by $Attack(\phi, \psi)$, is defined as follows: if $PreAttack(\phi, \psi) = \{\alpha\}$, then $Attack(\phi, \psi) = \alpha$; and otherwise $Attack(\phi, \psi) = null$.

In short, *PreAttack* is defined for any pair of axioms ϕ, ψ while *Attack* is defined for a pair of axioms ϕ, ψ for which $|PreAttack(\phi, \psi)| = 1$.

In Definition 2, if $PreAttack(\phi, \psi) = \{\alpha\}$ then either ψ or $\neg\psi$ is a base of ϕ. We say ψ ($\neg\psi$) is *attacked* by φ if ψ ($\neg\psi$) is a base of ϕ. In this sense, if $Attack(\phi, \psi) \neq null$ then there exists the *attack relation* between ϕ and ψ. The attack relation obeys a principle that there exists at most one base of ϕ (or φ) attacked by φ (or ϕ). However, *PreAttack* does not necessarily obey this principle.

As discussed above, the attack relation between two axioms can exactly capture their resolutive relation. Now, we are ready to define the *attack graph* of an ontology via the attack relation.

Definition 3 (Attack Graph)**.** Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ be an ontology. The *attack graph* for $\mathcal{O}$, denoted by $AttackGraph(\mathcal{O})$, is a graph $(\mathcal{V}, \mathcal{E})$ where each axiom in $\mathcal{O}$ is a vertex, that is, $\mathcal{V} = \mathcal{A} \cup \mathcal{T}$ and each edge $(\phi, label, \varphi) \in \mathcal{E}$ where $label = ID(Attack(\phi, \varphi))$ if $Attack(\phi, \varphi) \neq null$ for any ϕ, φ in $\mathcal{O}$. We denote $vertex(\mathcal{G}) = \mathcal{V}$ and $labels(\mathcal{G}) = \{label \mid (\phi, label, \varphi) \in \mathcal{E}\}$.

Intuitively, an attack graph is an undirected edge-labelled graph whose vertexes are axioms and two vertexes connected by an edge if and only if they are resolutive.

Each edge has a unique label since each $Attack(\phi, \psi)$ is either a singleton or an empty set. Besides, each edge has a label, either a named individual or individual variable x. Given an attack graph $\mathcal{G}$, $vertex(\mathcal{G})$ is also taken as an ontology. For instance, the attack graph $AttackGraph(\mathcal{O})$ of Pet ontology is shown in Figure 1.

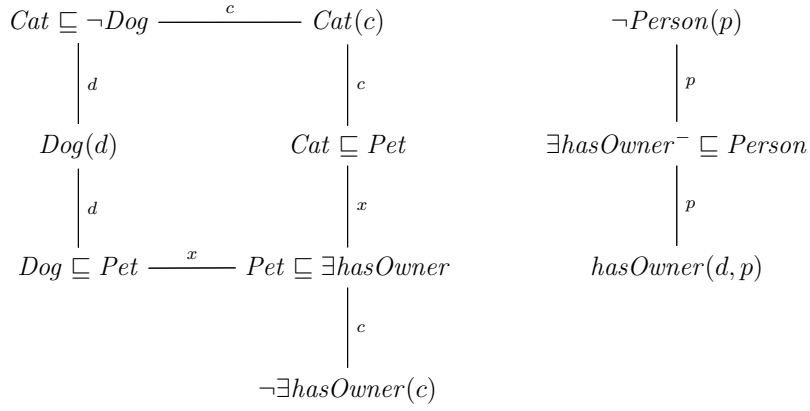

Figure 1. The attack graph of Pet ontology

The definitions of both *path*, *connected attack graph*, and *connected component* are standard.

4 SEARCHING AND GENERATING ARGUMENTS OVER ONTOLOGIES

In this section, we will introduce focal graphs within an attack graph to search all arguments and we then define support paths within a focal graph to generate all arguments for some axioms.

4.1 Searching Arguments Using Focal Graphs

In this subsection, we will introduce a focal graph in an attack graph to capture all supports of all arguments for some axioms.

Definition 4 (Closed vertex)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an attack graph. We define a *closed vertex* as follows:

- A vertex ϕ in form of $P(a, b)$ is *closed* with respect to $\mathcal{G}$ if there is a vertex $\psi \in \mathcal{V}$ such that $Attack(\phi, \psi) = \exists P(a)$ or $Attack(\phi, \psi) = \exists P^{-}(b)$;
- a vertex ϕ in form of $C(a)$ or $B \sqsubseteq C$ is *closed* with respect to $\mathcal{G}$ if for each base $\varphi \in \Gamma(\phi)$ there is a vertex $\psi \in \mathcal{V}$ such that $Attack(\phi, \psi) = \varphi$.

Intuitively speaking, each base of a closed vertex is attacked by other vertices. For instance, both vertices $Dog \sqsubseteq Pet$ and $Dog(d)$ are closed in the attack graph of Pet ontology shown in Figure 1.

Definition 5 (Closed graph)**.** Let $\mathcal{O}$ be an ontology. The *closed graph* of an ontology $\mathcal{O}$, denoted by $ClosedGraph(\mathcal{O})$, is the largest subgraph of $AttackGraph(\mathcal{O})$ whose each vertex is closed with respect to $\mathcal{G}$.

The closed graph allows us to focus on the relevant part of the ontology for constructing arguments.

For instance, in the Pet ontology $\mathcal{O}$ (shown in Example 1), let $\mathcal{O}^* = \mathcal{O} \cup \{Cat(d), Dog(c), \neg\exists hasOwner^{-}(p), \neg\exists hasOwner(d)\}$ be a sup-ontology of $\mathcal{O}$. Then $ClosedGraph(\mathcal{O}^*)$ is shown in Figure 2.

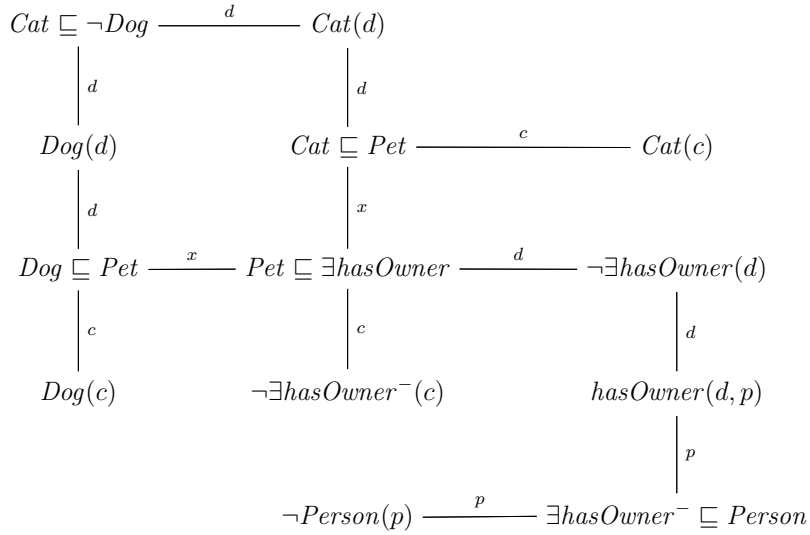

Figure 2. The closed graph of $\mathcal{O}^*$

Note that for an arbitrary closed graph $\mathcal{G}$, $vertex(\mathcal{G})$ is not necessarily inconsistent. For instance, let $\mathcal{T} = \{B_1 \sqsubseteq B_2, B_2 \sqsubseteq B_3, B_3 \sqsubseteq B_4, B_4 \sqsubseteq B_1\}$ be a TBox; we have $vertex(ClosedGraph(\mathcal{T})) = \mathcal{T}$ while $\mathcal{T}$ is consistent.

Arguments for an axiom are often related with several individuals. Thus, we only need to focus on some relevant labels (individuals) on a closed graph in order to further narrow the scope of search arguments.

Definition 6 (Focal graph)**.** Let $\mathcal{O}$ be an ontology and ϕ an axiom. The *focal graph* of ϕ in $\mathcal{O}$, denoted by $FocalGraph(\mathcal{O}, \phi)$, if it satisfies

- it is a connected subgraph of $ClosedGraph(\mathcal{O})$ containing vertex ϕ, that is, its each vertex is connected to its any other vertex by a path;
- the label of each edge is either $ID(\phi)$ or x.

Intuitively, a focal graph is a closed subgraph, whose edges are labeled by either "x" or a named individual.

For instance, in Pet ontology, the figure of $FocalGraph(\mathcal{O}^*, \neg\exists hasOwner(d))$ is shown in Figure 3. The figure of $FocalGraph(\mathcal{O}^*, \neg\exists hasOwner^-(c))$ is shown in Figure 4. Additionally, the figure of $FocalGraph(\mathcal{O}^*, \neg Person(p))$ is shown in Figure 5.

Though a focal graph $\mathcal{G}$ possibly contains some role assertion $R(a, b)$, for any vertices ϕ, ψ in $\mathcal{G}$, we have $Attacks(\phi, \Gamma(R(a, b))) = Attacks(\psi, \Gamma(R(a, b)))$.

Now, we collect vertices (axioms) on focal graphs.

Definition 7 (Zone)**.** Let $\mathcal{O}$ be an ontology and ϕ an axiom. The *zone* of ϕ in $\mathcal{O}$, denoted by $Zone(\mathcal{O}, \phi)$, is a set of axioms defined as follows:

- $Zone(\mathcal{O}, C(a)) = vertex(FocalGraph(\mathcal{O}', C(a)))$ where $\mathcal{O}' = \mathcal{O} \cup \{\neg C(a)\}$;
- $Zone(\mathcal{O}, B \sqsubseteq C) = vertex(FocalGraph(\mathcal{O}''), B(\tau)) \cup vertex(FocalGraph(\mathcal{O}''), \neg C(\tau))$; where τ is a new individual not occurring in $\mathcal{O}$ and $\mathcal{O}'' = \mathcal{O} \cup \{B(\tau), \neg C(\tau)\}$.

For instance, in Pet ontology, we have

- from Figure 3, $Zone(\mathcal{O}^*, \neg\exists hasOwner(d)) = \{Cat \sqsubseteq \neg Dog, Cat(d), Dog(d), Cat \sqsubseteq Pet, Dog \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner, \neg\exists hasOwner(d), hasOwner(d, p)\}$;
- from Figure 4, $Zone(\mathcal{O}^*, \neg\exists hasOwner(c)) = \{Dog(c), Dog \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner, Cat \sqsubseteq Pet, Cat(c), \neg\exists hasOwner(c)\}$;
- from Figure 5, $Zone(\mathcal{O}^*, \neg Person(p)) = \{\exists hasOwner^- \sqsubseteq Person, hasOwner(d, p)\}$.

The next result shows that zones contain exactly supports of all arguments.

Theorem 1. Let $\mathcal{O}$ be an ontology and ϕ an axiom. If $\langle \Phi, \phi \rangle$ is an argument for ϕ with respect to $\mathcal{O}$ then $\Phi \subseteq Zone(\mathcal{O}, \phi)$. That is, for any $\mathbf{A} \in Arg(\mathcal{O}, \phi)$, we have $Sup(\mathbf{A}) \subseteq Zone(\mathcal{O}, \phi)$.

Theorem 1 states that the zone of an axiom can characterize the scope of all arguments for it.

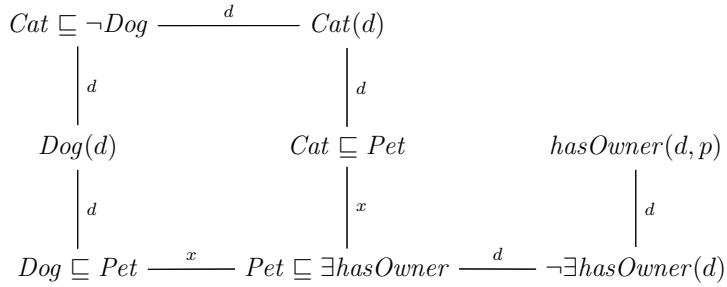

Figure 3. The focal graph of $\neg\exists hasOwner(d)$

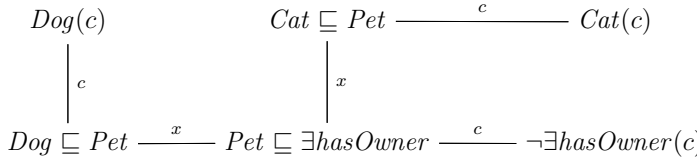

Figure 4. The focal graph of $\neg\exists hasOwner(c)$

4.2 Generating Arguments Using Support Paths

In this subsection, we will define support paths in a focal graph to generate all arguments for some axioms.

Let $\mathcal{O}$ be an ontology and ϕ an axiom. A *path* π is a sequence of vertices $\pi = \langle \phi_1, \ldots, \phi_n \rangle$ in $FocalGraph(\mathcal{O}, \phi)$ such that $Attacks(\phi_i, \phi_{i+1}) \neq null$ for any i $(1 \leq i \leq k-1)$. We denote $vertex(\pi) = \{\phi_1, \ldots, \phi_n\}$, that is, a set of all vertices occurring in π. In this sense, ϕ_i is the *predecessor* of ϕ_{i+1} and ϕ_{i+1} is the *successor* of ϕ_i. We denote $SubPath(\phi_i) = \langle \phi_i, \ldots, \phi_n \rangle$.

For instance, in $FocalGraph(\mathcal{O}, \neg\exists hasOwner(d))$ (shown in Figure 3), there are three paths π_i $(i = 1, 2, 3)$ starting at $\neg\exists hasOwner(d)$ where

- $\pi_1 = \langle \neg\exists hasOwner(d), Pet \sqsubseteq \exists hasOwner, Cat \sqsubseteq Pet, Cat(d), Cat \sqsubseteq \neg Dog, Dog(d) \rangle$;
- $\pi_2 = \langle \neg\exists hasOwner(d), Pet \sqsubseteq \exists hasOwner, Dog \sqsubseteq Pet, Dog(d) \rangle$;
- $\pi_3 = \langle \neg\exists hasOwner(d), hasOwner(d, p) \rangle$.

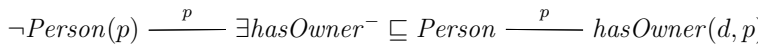

Figure 5. The focal graph of $\neg Person(p)$

Definition 8 (Complete path)**.** Let $\mathcal{O}$ be an ontology and ϕ an axiom. A path is *complete* in $\mathit{FocalGraph}(\mathcal{O}, \phi)$ if for any i $(2 \leq i \leq k-1)$, $\mathit{Attacks}(\phi_{i-1}, \phi_i) \neq \mathit{Attacks}(\phi_i, \phi_{i+1})$.

For instance, π_1 and π_2 are complete.

Note that each concept assertion only occurs in two endpoints of a complete path. For instance, the endpoints of both π_1 and π_2 are $\neg\exists \mathit{hasOwner}(d))$ and $\mathit{Dog}(d)$.

The idea in building a complete path $\pi = \langle \phi_1, \ldots, \phi_n \rangle$ is that in this way the set of axioms produced $\mathit{vertex}(\pi)$ such that at some step i, we have produced $\pi = \langle \phi_1, \ldots, \phi_i \rangle$, for any $\alpha \in \Gamma(\phi_i)) \setminus \Gamma(\phi_{i-1})$ (Because ϕ_1 is the endpoint of π, we have $|\Gamma(\phi_1)| = 1$ and let $\alpha = \phi_1$.) There is a node ϕ_{i+1} in $\mathit{FocalGraph}(\mathcal{O}, \phi)$ such that $\neg\alpha \in \Gamma(\phi_{i+1})$ and the set $\{\phi_1, \ldots, \phi_i, \phi_{i+1}\} \cup \{\neg\alpha\}$ is inconsistent. Thereby, we can ensure that $\{\phi_1, \ldots, \phi_i\} \models \alpha$.

For instance, $\mathit{vertex}(\pi_1) \setminus \{\neg\exists \mathit{hasOwner}(d)\} \models \exists \mathit{hasOwner}(d)$ and $\mathit{vertex}(\pi_2) \setminus \{\mathit{Dog}(d)\} \models \neg \mathit{Dog}(d)$.

Note that if a path π contains some role assertion $R(a, b)$, then there exists exactly vertex ϕ connecting $R(a, b)$ since either $\mathit{Attacks}(\psi, R(a, b)) = \{\exists R(a)\}$ for any ψ holds or $\mathit{Attacks}(\psi, R(a, b)) = \{\exists R^-(b)\}$ for any ψ holds.

For instance, a path $\langle \mathit{hasOwner}(d, p), \exists \mathit{hasOwner}^- \sqsubseteq \mathit{Person}, \neg \mathit{Person}(p) \rangle$ is complete and $\mathit{Attacks}(\mathit{hasOwner}(d, p), \exists \mathit{hasOwner}^- \sqsubseteq \mathit{Person}) = \{\exists \mathit{hasOwner}^-(p)\}$ in $\mathit{FocalGraph}(\mathcal{O}, \neg \mathit{Person}(p))$ (shown in Figure 4).

Next, we will introduce two features of complete paths, namely, *consistency* and *minimality*.

Definition 9 (Consistent path)**.** Let $\mathcal{O}$ be an ontology and ϕ an axiom. A complete path π is *consistent* in $\mathit{FocalGraph}(\mathcal{O}, \phi)$ if for any vertices ψ and φ, if ψ' is the predecessor of ψ and φ' is the predecessor of φ', $\mathit{Attacks}(\psi, \psi') \neq \mathit{Attacks}(\varphi, \varphi')$.

For instance, because $\mathit{Attacks}(\mathit{Cat} \sqsubseteq \mathit{Pet}, \mathit{Cat}(d)) = \mathit{Attacks}(\mathit{Cat}(d), \mathit{Cat} \sqsubseteq \neg \mathit{Dog}) = \{\mathit{Cat}(d)\}$, π_2 is consistent while π_1 is not consistent.

Indeed, the consistency of complete paths can avoid the redundancy of edges.

Definition 10 (Minimal path)**.** Let $\mathcal{O}$ be an ontology and ϕ an axiom. A complete path π is *minimal* in $\mathit{FocalGraph}(\mathcal{O}, \phi)$ if there exists no any complete path π' in $\mathit{FocalGraph}(\mathcal{O}, \phi)$ such that $\mathit{vertex}(\pi') \subset \mathit{vertex}(\pi)$.

Indeed, the minimality of complete paths can avoid the redundancy of vertices.

For instance, π_2 is minimal while π_1 is not minimal since there exists a path $\pi_4 = \langle \neg\exists \mathit{hasOwner}(d_1)), \mathit{Pet} \sqsubseteq \exists \mathit{hasOwner}, \mathit{Cat} \sqsubseteq \mathit{Pet}, \mathit{Cat}(d) \rangle$ such that $\mathit{vertex}(\pi_4) \subset \mathit{vertex}(\pi_1)$ and π_4 is complete and consistent.

Now, we are ready to introduce support paths for some concept assertions.

Definition 11 (Support path)**.** Let $\mathcal{O}$ be an ontology and ϕ an axiom. A *support path* π for ϕ with respect to $\mathcal{O}$ is a complete path containing a vertex ϕ in $\mathit{FocalGraph}(\mathcal{O}, \phi)$ that is minimal and consistent.

For instance, π_2 and π_4 are support paths for $\neg\exists hasOwner(d))$ with respect to $\mathcal{O}$.

Let π be a support path. For any $u \in vertex(\pi)$, we denote

$$LiteralsRes(\psi) = Literals(\psi) \setminus \{Attacks(\varphi, \varphi') \mid \varphi, \varphi' \in SubtPath(\psi))\};$$

where

$$Literals(\psi) = \bigcup_{\varphi \in SubPath(\psi)} \Gamma(\varphi).$$

Theorem 2. Let $\mathcal{O}$ be an ontology and $C(a)$ a concept assertion. If π is a complete support path for $\neg C(a)$ with respect to $\mathcal{O}$, then $\langle \Phi, C(a)\rangle$ with $\Phi = vertex(\pi) \setminus \{\neg C(a)\}$ is an argument.

For instance, let $\Phi_2 = vertex(\pi_2) \setminus \{\neg\exists hasOwner(d)\}$ and $\Phi_1 = vertex(\pi_3) \setminus \{\neg\exists hasOwner(d)\}$; it obviously concludes that $\langle \Phi_2, \exists hasOwner(d)\rangle$ and $\langle \Phi_1, \exists hasOwner(d)\rangle$ are arguments for $\exists hasOwner(d)$ with respect to $\mathcal{O}$.

Analogously, we find that $\langle \neg\exists hasOwner(c), Pet \sqsubseteq \exists hasOwner, Cat \sqsubseteq Pet, Cat(c)\rangle$ is a support path for $\neg\exists hasOwner(c)$ in Figure 4 and $\langle hasOwner(d, p), \exists hasOwner^- \sqsubseteq Person, \neg Person(p)\rangle$ is a support path for $Person(p)$ in Figure 5. We exactly generate arguments $\mathbf{A}_3$ and $\mathbf{A}_4$ in Example 1.

Next, we consider how to generate argument for concept inclusion by using support paths. Note that the support of an argument for a concept inclusion is a subset of a TBox. In the following, we mainly consider TBoxes instead of ontologies.

Theorem 3. Let $\mathcal{T}$ be a TBox and $C \sqsubseteq D$ a concept inclusion. Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ where $\mathcal{A} = \{C(\tau), \neg D(\tau)\}$ where τ is a new individual name. If π is either a support path for $C(\tau)$ with respect to $\mathcal{O}$ or a support path for $\neg D(\tau)$ with respect to $\mathcal{O}$ then $\langle \Phi, C \sqsubseteq D\rangle$ with $\Phi = vertex(\pi) \setminus \{C(\tau), \neg D(\tau)\}$ is an argument.

For instance, let $\mathcal{O}' = (\mathcal{T}, \{Dog(\tau), \neg\exists hasOwner(\tau)\})$ be an ontology. The focal graph $FocalGraph(\mathcal{O}, Dog(\tau))$ is shown in Figure 6. There exists a complete support path $\langle Dog(\tau), Dog \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner, \neg\exists hasOwner(\tau)\rangle$. Thus the argument for $Dog \sqsubseteq \exists hasOwner$ is $\langle(\{Dog \sqsubseteq Pet, Pet \sqsubseteq \exists hasOwner\}, \emptyset), Dog \sqsubseteq \exists hasOwner\rangle$, i.e., $\mathbf{A}_5$.

$$Dog(\tau) \overset{\tau}{\text{———}} Dog \sqsubseteq Pet \overset{x}{\text{———}} Pet \sqsubseteq \exists hasOwner \overset{\tau}{\text{———}} \neg\exists hasOwner(\tau)$$

Figure 6. The focal graph of $Dog(\tau)$

In the rest of this section, we discuss the computational complexity of generating arguments over DL-Lite$_{core}$ ontologies.

Given an ontology $\mathcal{O}$ in DL-Lite$_{core}$, if we assume that $|\mathcal{O}| = n$ and for any two axioms ϕ, ψ, the computational time of $PreAttack(\phi, \psi)$ is 1, then the time of constructing $AttackGraph(\mathcal{O})$ is $4n^2$, the time of constructing $ClosedGraph(\mathcal{O})$ is

$8n^2$ and the time of constructing $\mathit{FocalGraph}(\mathcal{O}, \phi)$ some axiom ϕ in $\mathcal{O}$ is $8n^2 + 4n^4$ respectively. In other words, $\mathit{AttackGraph}(\mathcal{O})$, $\mathit{ClosedGraph}(\mathcal{O})$ can be constructed in $O(n^2)$ and $\mathit{FocalGraph}(\mathcal{O}, \phi)$ can be constructed in $O(n^4)$.

In short, the attack graph and focal graph are polynomial to the size of the ontology. Moreover, the focal graph contains at most n nodes and n^2 edges. The satisfiability problem for DL-Lite$_{core}$ is in polynomial time [1], which is the same as the satisfiability problem of propositional logic. In propositional logic, the problem of generating arguments is in the second level of the polynomial hierarchy where the problem is considered as the abduction problem [23]. Thus the complexity of generating support paths is not higher than the complexity of that in propositional revision. Therefore, the complexity of our approach to constructing focal graphs and generating arguments over DL-Lite$_{core}$ ontologies is in the second level of the polynomial hierarchy.

5 EXAMPLE: A PRACTICAL ONTOLOGY

In this section, we apply the proposal approach to discuss a practical ontology, so-called *buggyPolicy* ontology ([2]), whose incoherency is caused by over-defining.

The buggyPolicy ontology can be equivalently written as a DL-Lite$_{core}$ ontology which contains 22 axioms ψ_i as follows:

$$
\begin{aligned}
\psi_1 &: \mathit{ExactlyOneExamplePolicy} \sqsubseteq \mathit{Policy};\\
\psi_2 &: \mathit{GeneralReliabilityKerberosPolicy} \sqsubseteq \mathit{Policy};\\
\psi_3 &: \mathit{GeneralReliabilityUserPolicy} \sqsubseteq \mathit{Reliable};\\
\psi_4 &: \mathit{GeneralReliabilityUserPolicy} \sqsubseteq \mathit{UserToken};\\
\psi_5 &: \mathit{GeneralReliabilityUserPolicy} \sqsubseteq \mathit{Policy};\\
\psi_6 &: \mathit{GeneralReliabilityUserPolicy} \sqsubseteq \neg\mathit{Messaging};\\
\psi_7 &: \mathit{X509} \sqsubseteq \mathit{SecurityTokenType};\\
\psi_8 &: \mathit{IncoherentPolicy} \sqsubseteq \mathit{Policy};\\
\psi_9 &: \mathit{Kerberos} \sqsubseteq \mathit{SecurityTokenType};\\
\psi_{10} &: \mathit{Kerberos} \sqsubseteq \neg\mathit{Messaging};\\
\psi_{11} &: \mathit{Reliable} \sqsubseteq \mathit{Messaging};\\
\psi_{12} &: \mathit{RetryOnFailureUserPolicy} \sqsubseteq \mathit{Policy};\\
\psi_{13} &: \mathit{RetryUntilSucceedUserPolicy} \sqsubseteq \mathit{Policy};\\
\psi_{14} &: \mathit{RetryOnFailure} \sqsubseteq \mathit{Reliable};\\
\psi_{15} &: \mathit{RetryUntilSucceed} \sqsubseteq \mathit{Reliable};\\
\psi_{16} &: \mathit{UserToken} \sqsubseteq \mathit{SecurityTokenType};\\
\psi_{17} &: \mathit{RetryOnFailureUserPolicy} \sqsubseteq \mathit{RetryOnFailure};\\
\psi_{18} &: \mathit{RetryOnFailureUserPolicy} \sqsubseteq \mathit{UserToken};\\
\psi_{19} &: \mathit{RetryUntilSucceedUserPolicy} \sqsubseteq \mathit{UserToken};\\
\psi_{20} &: \mathit{RetryUntilSucceedUserPolicy} \sqsubseteq \mathit{RetryUntilSucceed};\\
\psi_{21} &: \mathit{IncoherentPolicy} \sqsubseteq \mathit{RetryOnFailureUserPolicy};\\
\psi_{22} &: \mathit{IncoherentPolicy} \sqsubseteq \mathit{RetryUntilSucceedUserPolicy}.
\end{aligned}
$$

Note that $\{\psi_3, \psi_{11}\}$ implies $\mathit{GeneralReliabilityUserPolicy} \sqsubseteq \mathit{Messaging}$, which conflicts with ψ_6. Thus $\mathit{GeneralReliabilityUserPolicy}$ is unsatisfiable.

Now, we add two axioms $\psi_{23} = \mathit{GeneralReliabilityUserPolicy}(id)$ and $\psi_{24} = \mathit{IncoherentPolicy}(id)$, where id is an individual, and then we obtain a new ontology $\mathcal{O}_p$ which is inconsistent because $\mathit{Messaging}(id)$ and $\neg\mathit{Messaging}(id)$ conflict with each other. The attack graph of $\mathcal{O}_p$ is shown in Figure 7.

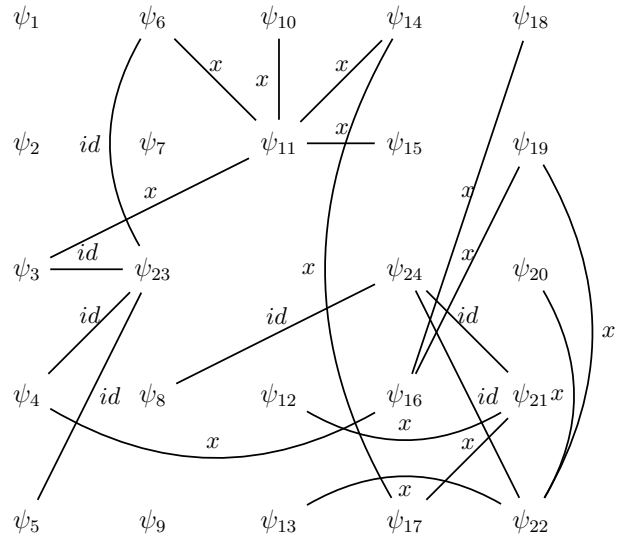

Figure 7. The attack graph of $\mathcal{O}_p$

Next, we will apply the proposal approach to generating all arguments for four following axioms in $\mathcal{O}_p$:

α_1 : $\mathit{Messaging}(id)$; α_2 : $\mathit{Reliable}(id)$;
α_3 : $\mathit{Policy}(id)$; α_4 : $\mathit{RetryUntilSucceed}(id)$.

Firstly, let $\mathcal{O}_p^i = \mathcal{O}_p \cup \{\neg\alpha_i\}$ $(i = 1, 2, 3, 4)$.

- The focal graph of $\neg\mathit{Messaging}(id)$ in $\mathcal{O}_p^1$ is shown in Figure 8. There exist two complete support paths $\pi_{11} = \langle\neg\alpha_1, \psi_{11}, \psi_3, \psi_{23}\rangle$, $\pi_{12} = \langle\neg\alpha_1, \psi_{11}, \psi_6, \psi_{23}\rangle$. Thus, there exist two arguments for α_1, $\langle(\{\psi_{11}, \psi_3\}, \{\psi_{23}\}), \alpha_1\rangle$ and $\langle(\{\psi_{11}, \psi_6\}, \{\psi_{23}\}), \alpha_1\rangle$.
- The focal graph of $\neg\mathit{Reliable}(id)$ in $\mathcal{O}_p^2$ is shown in Figure 9. There exists a complete support path $\pi_{21} = \langle\neg\alpha_2, \psi_3, \psi_{23}\rangle$. Thus, there exists an argument for α_2, $\langle(\{\psi_3\}, \{\psi_{23}\}), \alpha_2\rangle$.
- The focal graph of $\neg\mathit{Policy}(id)$ in $\mathcal{O}_p^3$ is shown in Figure 10. There exist three complete support paths $\pi_{31} = \langle\neg\alpha_2, \psi_5, \psi_{23}\rangle$, $\pi_{32} = \langle\neg\alpha_2, \psi_8, \psi_{24}\rangle$, $\pi_{33} =$

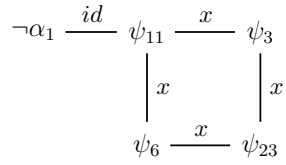

Figure 8. The focal graph of $\neg Messaging(id)$ in $\mathcal{O}_p^1$

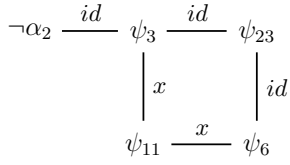

Figure 9. The focal graph of $\neg Reliable(id)$ in $\mathcal{O}_p^2$

$\langle \neg\alpha_2, \psi_{13}, \psi_{22}, \psi_{24} \rangle$. Thus, there exist three arguments for α_3, $\langle (\{\psi_5\}, \{\psi_{23}\}), \alpha_3 \rangle$, $\langle (\{\psi_8\}, \{\psi_{24}\}), \alpha_3 \rangle$ and $\langle (\{\psi_{13}, \psi_{22}\}, \{\psi_{24}\}), \alpha_3 \rangle$.

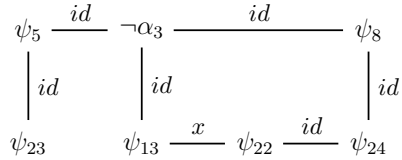

Figure 10. The focal graph of $\neg Policy(id)$ in $\mathcal{O}_p^3$

- The focal graph of $\neg RetryUntilSucceed(id)$ in $\mathcal{O}_p^4$ is shown in Figure 11. There exists a complete support path $\pi_{41} = \langle \neg\alpha_4, \psi_{20}, \psi_{22}, \psi_{24} \rangle$. Thus there exists an argument for α_4, $\langle (\{\psi_{20}, \psi_{22}\}, \{\psi_{24}\}), \alpha_4 \rangle$.

Based on this example, we conclude that the approach proposed in this paper is feasible in generating all arguments over practical ontologies.

6 DISCUSSION

Argumentation is the interdisciplinary study of how conclusions can be reached through logical reasoning [13]. One important application of argumentation is reasoning with knowledge bases containing some conflicts. Recently, there are some proposals for logic-based argumentation in reasoning with inconsistent DL ontologies [4, 14, 27, 6, 15, 28]. The common idea behind of them is employing some dialogue mechanisms to judge which of the knowledge causing inconsistency is true such as [14, 6, 15] based on Dung's graph-based dialogue [13] and [27, 28] based

$$\neg\alpha_4 \overset{id}{\text{———}} \psi_{20} \overset{x}{\text{———}} \psi_{22} \overset{id}{\text{———}} \psi_{24}$$

Figure 11. The focal graph of $\neg RetryUntilSucceed(id)$ in $\mathcal{O}_p^4$

on Besnard and Hunter's tree-based dialogue mechanism [12]. However, for argumentation, it is computationally challenging to generate arguments from ontologies. Indeed, generating arguments for propositional axioms is seeking the existence of a minimal subset of a set of propositional axioms that implies the consequent. In other words, this problem can be considered as abduction problem which is in the second level of the polynomial hierarchy [19]. As mentioned in [23], the difficult nature of argumentation has been underlined by studies concerning the complexity of finding individual arguments [16]. Efstathiou and Hunter successfully developed an approach to searching arguments by using connection graphs [21] and generating arguments [23] by using support trees for propositional axioms.

In this paper, inspiring from the approach proposed by Efstathiou and Hunter, we have presented an approach, which bring no higher complexity than that of the EH's approach to generating arguments for propositional axioms, to search and generate arguments for DL axioms over DL-Lite ontologies. Compared with Efstathiou and Hunter's method, our approach has the following improvements:

- we define attack graph as an edge-labeled connection graph whose labels can characterize individuals and edges can characterize three kinds of axioms (concept inclusions, concept assertions and role assertions) by refining resolutive conditions;
- we refine the closed condition so that role assertions can be characterized and restrict focal graphs by its labels so that the searching scope of arguments can be accurate;
- we present support paths and the completeness of support paths to characterize the support sets of arguments instead of support trees in [23] so that we can precisely generate arguments over DL-Lite ontologies.

Note that our proposal approach is based on the condition that an ontology can be embedded into a graph where axioms are taken as nodes and the resolutive relation between axioms is taken as a set of edges between them. However, unlike DL-Lite, the resolutive relation between axioms in expressive DLs such as $\mathcal{ALC}$ is not apparently characterized by their syntactical structures. There will be some challenges to adapt our proposal approach to generate arguments over expressive DL ontologies. In future work, we will improve our proposal approach for expressive DLs. Besides, we will consider to implement a system based on JArgue for propositional axioms implemented by [22], which is not an open source so far, as a future work.

Acknowledgments

We would like to thank the anonymous referees for their critical comments which helped us to improve the paper. Xiaowang Zhang is funded by the project of Research Foundation Flanders under grant G. 0489.10N and Zuoquan Lin is funded by the program of the National Natural Science Foundation of China (NSFC) under grant 60973003.

A APPENDIX: PROOFS

Proposition 1. Let ϕ and ψ be two axioms. The pair (ϕ, ψ) is resolutive if and only if $\{\phi, \psi\}$ contains a conflict.

Proof. Firstly, we prove the "if" direction. By the definition of resolution rules, the preconditions of **rule-1**, **rule-3** and **rule-4** contain conflicts in form of $\{A(t), \neg A(t)\}$ and the precondition **rule-2** contains conflicts in form of $\{\exists R(t), \neg\exists R(t)\}$.

We then prove the "only if" direction. By the definition of conflicts in two forms of: $\{A(t), \neg A(t)\}$ and $\{\exists R(t), \neg\exists R(t)\}$. If $\{\phi, \psi\}$ contains conflict $\{A(t), \neg A(t)\}$ then it meets the preconditions of **rule-1**, **rule-3** and **rule-4**. If $\{\phi, \psi\}$ contains conflict $\{\exists R(t), \neg\exists R(t)\}$ then it meets the precondition of **rule-2**. That is, $\{\phi, \psi\}$ can use one of resolution rules. Therefore, $\{\phi, \psi\}$ is resolutive. □

Proposition 2. Let ϕ, ψ be two axioms. $\mathit{PreAttack}(\phi, \psi) \neq \emptyset$ if and only if the pair (ϕ, ψ) is resolutive.

Proof. Firstly, we prove the "if" direction. if the pair (ϕ, ψ) is resolutive then they contain a conflict $\{\alpha, \neg\alpha\}$ and $\alpha \in \Gamma(\phi)$, $\neg\alpha \in \Gamma(\psi)$ by Proposition 1. Thus either $\alpha \in \mathit{PreAttack}(\phi, \psi)$ or $\neg\alpha \in \mathit{PreAttack}(\phi, \psi)$. We conclude that $\mathit{PreAttack}(\phi, \psi) \neq \emptyset$. We then prove the "only if" direction. If $\mathit{PreAttack}(\phi, \psi) \neq \emptyset$ then, without loss of generality, there exists a conflict $\{\alpha, \neg\alpha\}$ and $\alpha \in \Gamma(\phi)$, $\neg\alpha \in \Gamma(\psi)$. Thus the pair (ϕ, ψ) is resolutive by Proposition 1. □

Theorem 1. Let $\mathcal{O}$ be an ontology and ϕ an axiom. If $\langle\Phi, \phi\rangle$ is an argument for ϕ with respect to $\mathcal{O}$ then $\Phi \subseteq \mathit{Zone}(\mathcal{O}, \phi)$. That is, for any $\mathbf{A} \in \mathit{Arg}(\mathcal{O}, \phi)$, we have $\mathit{Sup}(\mathbf{A}) \subseteq \mathit{Zone}(\mathcal{O}, \phi)$.

Proof. Let $\mathbf{A} = \langle\Phi, \phi\rangle \in \mathit{Arg}(\mathcal{O}, \phi)$. $\Phi \models \phi$ and Φ is consistent. We consider three cases of ϕ as follows:

- if $\phi = C(a)$, then $\Phi \cup \{\neg C(a)\}$ is inconsistent. Then a contradiction is obtained by exhaustedly applying resolution rules by Lemma 2. We need to show that $\mathit{vertex}(\mathit{FocalGraph}(\Phi \cup \{\neg C(a)\})) = \Phi \cup \{\neg C(a)\}$ in two steps by Definition 7:
  - We need to show $\mathit{vertex}(\mathit{ClosedGraph}(\Phi \cup \{\neg C(a)\})) = \Phi \cup \{\neg C(a)\}$, that is, for any vertex ψ, each base $\varphi \in \Gamma(\psi)$ is attacked by Definition 2. Assume that $\mathit{vertex}(\mathit{ClosedGraph}(\Phi \cup \{\neg C(a)\})) \neq \Phi \cup \{\neg C(a)\}$, that is, there exists

a vertex ψ' *AttackGraph*$(\Phi \cup \{\neg C(a)\})$ such that a base $\varphi' \in \Gamma(\psi')$ can not be attacked. Then φ' is retained after exhaustedly applying resolution rules. It contradicts with Lemma 2.

– We need to show *ClosedGraph*$(\Phi \cup \{\neg C(a)\}) =$ *FocalGraph*$(\Phi \cup \{\neg C(a)\})$, that is, each edge on *ClosedGraph*$(\Phi \cup \{\neg C(a)\})$ has two labels x and a. Because $\Gamma(P(a,b)) = \{\exists P(a), \exists P^-(b)\}$, if $\exists P(a)$ is attacked then $P(a,b)$ is *FocalGraph*$(\Phi \cup \{\neg C(a)\})$. Each edge has either label x or label a on *ClosedGraph*$(\Phi \cup \{\neg C(a)\})$.

- if $\phi = P(a,b)$ or $P^-(b,a)$, then $\Phi = \{P(a,b)\}$. Then $P(a,b) \in$ *Zone*$(\mathcal{O}, \phi)$ by Definition 7.
- if $\phi = B \sqsubseteq C$, then $\Phi \cup \{B(\tau), \neg C(\tau)\}$ is inconsistent by Lemma 1 and Φ only contains concept inclusions. In *AttackGraph*$(\Phi \cup \{B(\tau), \neg C(\tau)\})$, each edge has either label x or label τ. Similar to the proof (1), for any vertex $\psi \in \Phi \cup \{B(\tau), \neg C(\tau)\}$, each base $\varphi \in \Gamma(\psi)$ is attacked since $\Phi \cup \{B(\tau), \neg C(\tau)\}$ is inconsistent by Lemma 2. We can conclude that *vertex*(*ClosedGraph*$(\Phi \cup \{B(\tau), \neg C(\tau)\})) \subseteq$ *vertex*(*FocalGraph*$(\Phi \cup \{B(\tau), \neg C(\tau)\}, B(\tau))) \cup$ *vertex* (*FocalGraph*$(\Phi \cup \{B(\tau), \neg C(\tau)\}, \neg C(\tau)))$.

Therefore, we can conclude that $\Phi \subseteq$ *Zone*$(\mathcal{O}, \phi)$. □

Theorem 2. Let $\mathcal{O}$ be an ontology and $C(a)$ a concept assertion. If π is a complete support path for $\neg C(a)$ with respect to $\mathcal{O}$, then $\langle \Phi, C(a) \rangle$ with $\Phi =$ *vertex*$(\pi) \setminus \{\neg C(a)\}$ is an argument.

Proof. Because, in *FocalGraph*$(\mathcal{O} \cup \{\neg C(a)\}, \neg C(a))$, π is a complete support path for $C(a)$ with respect to $\mathcal{O}$, $\neg C(a)$ must occur in the endpoint of π since $\Gamma(\neg C(a))$ is a singleton, that is, there exists at most one edge in π. Without loss of generality assume that $\pi = \langle \neg C(a), \phi_1, \ldots, \phi_n \rangle$. That is, ϕ_1 be the successor of $\neg C(a)$. Then $\Phi =$ *vertex*$(\pi) \setminus \{\neg C(a)\}$.

- $\Phi \models C(a)$. Let $\pi^{-1} = \langle \phi_n, \ldots, \phi_1, \neg C(a) \rangle$. Because π is a complete path, we conclude that π^{-1} is a complete path by Definition 8. In π^{-1}, the vertex $\neg C(a)$ can be added behind of the path $\langle \phi_n, \ldots, \phi_1 \rangle$. Based on the way of building complete paths, $\{\phi_n, \ldots, \phi_1\} \cup \{\neg C(a)\}$ is inconsistent. Thus $\{\phi_n, \ldots, \phi_1\} \models C(a)$. Therefore, $\Phi \models C(a)$ since $\Phi = \{\phi_1, \ldots, \phi_n\}$.
- Φ is consistent. Assume that Φ is inconsistent. Then we have *LiteralsRes*$(\phi_1) = \emptyset$. Thus there exist some vertices ϕ_i, ϕ_j such that *Attacks*$(\phi_i, \phi_j) = \neg C(a)$ which contradicts the fact that π is consistent by Definition 9.
- There exists no $\Phi' \subset \Phi$ such that $\Phi \models C(a)$. Assume that there exists some $\Phi' \subset \Phi$ such that $\Phi' \models C(a)$. Without loss of generality assume Φ' is minimal for entailing $C(a)$. Thus there exists a support path π' such that $\Phi' =$ *vertex*$(\pi') \setminus \{\neg C(a)\}$ by Definition 10. It is not hard to show that π' is a complete path by Definition 8. Then, by Definition 11, π does not satisfy the definition for

a minimal complete path which contradicts the precondition that π is a support path.

□

Theorem 3. Let $\mathcal{T}$ be a TBox and $C \sqsubseteq D$ a concept inclusion. Let $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ where $\mathcal{A} = \{C(\tau), \neg D(\tau)\}$ where τ is a new individual name. If π is either a support path for $C(\tau)$ with respect to $\mathcal{O}$ or a support path for $\neg D(\tau)$ with respect to $\mathcal{O}$ then $\langle \Phi, C \sqsubseteq D \rangle$ with $\Phi = vertex(\pi) \setminus \{C(\tau), \neg D(\tau)\}$ is an argument.

Proof. We will prove this theorem in three cases of π:

- π is a support path for $C(\tau)$ but not a support path for $\neg D(\tau)$ with respect to $\mathcal{O}$. $C(\tau)$ must be one of endpoints of π. By Theorem 2, $\langle \Phi, \neg C(\tau) \rangle$ is an argument. Thus $\Phi \models \neg C(\tau)$. Because $\Phi \subseteq \mathcal{T}$, that is, a set of some concept inclusions, $\Phi \models C \sqsubseteq \bot$. Then $\Phi \models C \sqsubseteq D$.
- π is a support path for $\neg D(\tau)$ but not a support path for $C(\tau)$ with respect to $\mathcal{O}$. $\neg D(\tau)$ must be one of endpoints of π. By Theorem 2, $\langle \Phi, D(\tau) \rangle$ is an argument. Thus $\Phi \models D(\tau)$. Because $\Phi \subseteq \mathcal{T}$, that is, a set of some concept inclusions, $\Phi \models \top \sqsubseteq D$. Then $\Phi \models C \sqsubseteq D$.
- π is a support path for both $C(\tau)$ and $\neg D(\tau)$ with respect to $\mathcal{O}$. $C(\tau), \neg D(\tau)$ must be exactly two endpoints of π. By Theorem 2, $\langle \Phi, \neg C(\tau) \rangle$ and $\langle \Phi, D(\tau) \rangle$ are two arguments. Thus $\Phi \models \neg C \sqcup D(\tau)$. By the selection of τ, we conclude that for any individual name x, we have $\Phi \models \neg C \sqcup D(x)$ since $\Phi \subseteq \mathcal{T}$. Therefore, we conclude that $\Phi \models C \sqsubseteq D$.

□

REFERENCES

[1] Artale, A.—Calvanese, D.–Kontchakov, R.—Zakharyaschev, M.: The DL-Lite Family and Relations. J. Artif. Intell. Res. 36, 2009 pp. 1–69.

[2] Kalyanpur, A.—Parsia, B.—Sirin, E.—Grau, C. B.: Debugging Unsatisfiable Classes in OWL Ontologies. J. Web Sem., Vol. 3, 2006, No. 4, pp. 268–293.

[3] Motik, B.: Reasoning in Description Logics Using Resolution and Deductive Databases. Ph. D. Thesis, Karlsruhe University 2006.

[4] Tempich, C.—Simperl, E. P. B.—Luczak, M.—Studer, R.—Pinto, H. S.: Argumentation-Based Ontology Engineering. IEEE Intel. Sys., Vol. 22, 2007, No. 6, pp. 52–59.

[5] Calvanese, D.—Giacomo, G. D.—Lembo, D.—Lenzerini, M.—Rosati, R.: Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family. J. Autom. Reasoning, Vol. 39, 2007, No. 3, pp. 385–429.

[6] Black, E.—Hunter, A.—Pan, J. Z.: An Argument-Based Approach to Using Multiple Ontologies. In: L. Godo, A. Pugliese (Eds.): Proc. 3rd Int. Conf. Scalable Uncertainty Management, Washington, DC 2009, pp. 68–79.

[7] Paraiso, E. C.—Malucelli, A.: Ontologies Supporting Intelligent Agent-Based Assistance. Comput. Inform., Vol. 30, 2011, No. 4, pp. 829–855.

[8] Baader, F.—Calvanese, D.—McGuinness, D. L.—Nardi, D.—Patel-Schneider, P. F.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, UK 2003.

[9] Flouris, G.—Huang, Z.—Pan, J. Z.—Plexousakis, D.—Wache, H.: Inconsistencies, Negations and Changes in Ontologies. Proc. 21st National Conf. Artif. Intell., USA 2006, pp. 1295–1300.

[10] Horrocks, I.: Ontologies and the Semantic Web. Comm. ACM, Vol. 51, 2008, No. 12, pp. 58–67.

[11] Bertossi, L. E.—Hunter, A.—Schaub, T.: Inconsistency Tolerance. Springer 2005.

[12] Besnard, P.—Hunter, A.: A Logic-Based Theory of Deductive Arguments. Artif. Intell., Vol. 128, 2001, No. 1-2, pp. 203–235.

[13] Dung, P. M.: On the Acceptability of Arguments and Its Fundamental Role in Nonmonotonic Reasoning. Logic Programming and n-Person Games. Artif. Intell., Vol. 77, 1995, No. 2, pp. 321–358.

[14] Gómez, S. A.—Chesñevar, C. I.—Simari, G. R.: An Argumentative Approach to Reasoning with Inconsistent Ontologies. Proc. 1st Knowledge Representation Ontology Workshop, T. Meyer and M. A. Orgun (Eds)., Australia 2008, pp. 11–20.

[15] Gómez, S. A.—Chesñevar, S. I.—Simari, G. R.: Reasoning with Inconsistent Ontologies Through Argumentation. Applied Artif. Intell., Vol. 24, 2010, No. 1-2, pp. 102–148.

[16] Parsons, S.—Wooldridge, M.—Amgoud, L.: Properties and Complexity of Some Formal Inter-agent dialogues. J. Log. Comp., Vol. 13, 2003, No. 3, pp. 347–376.

[17] Schlobach, S.—Cornet, R.: Non-Standard Reasoning Services for the Debugging of Description Logic Terminologies. Proc. 18th Int. Joint Conf. Artif. Intell., G. Gottlob and T. Walsh (Eds.), Acapulco 2003, pp. 355–362.

[18] Berners-Lee, T.—Hendler, J.—Lassila, O.: The Semantic Web: Scientific American. Scientific American Magazine, USA 2001.

[19] Eiter, T.—Gottlob, G.: The Complexity of Logic-based Abduction. J. ACM, Vol. 42, 1995, pp. 3–42.

[20] Meyer, T.—Lee, K.—Booth, R.: Knowledge Integration for Description Logics. Proc. 20th National Conf. Artif. Intell., M. Veloso and S. Kambhampati (Eds.), Pennsylvania 2005, pp. 645–650.

[21] Efstathiou, V.—Hunter, A.: Focused Search for Arguments from Propositional Knowledge. Proc. 2nd Int. Conf. Computational Models of Argument, P. Besnard, S. Doutre and A. Hunter (Eds.), Toulouse 2008, pp. 159–170.

[22] Efstathiou, V.—Hunter, A.: JArgue: An Implemented Argumentation system for classical Propositional Logic. Online demo 2010.

[23] Efstathiou, V.—Hunter, A.: Algorithms for Generating Arguments and Counterarguments in Propositional Logic. Int. J. Approx. Reasoning, Vol. 52, 2011, pp. 672–704.

[24] Funika, W.—Godowski, P.—Pegiel, P.—Król, D.: Semantic-Oriented Performance Monitoring of Distributed Applications. Comput. Inform., Vol. 31, 2012, No. 2, pp. 427–446.

[25] Zhang, X.—Lin, Z.: An Argumentation-Based Approach to Handling Inconsistencies in DL-Lite. Proc. 32nd Ann. German Conf. Artif. Intell., B. Mertsching, M. Hund and M. Zaheer Aziz (Eds)., Paderborn 2009, pp. 615–622.

[26] Zhang, X.—Xiao, G.—Lin, Z.: A Tableau Algorithm for Handling Inconsistency in OWL. Proc. 6th European Semantic Web Conf., L. Aroyo et al. (Eds.), Heraklion 2009, pp. 399–413.

[27] Zhang, X.—Zhang, Z.—Lin, Z.: An Argumentative Semantics for Paraconsistent Reasoning in Description Logic ALC. Proc. 22nd Int. Workshop Description Logics, B. Cuenca Grau, I. Horrocks, B. Motik and U. Sattler (Eds.), Oxford 2009.

[28] Zhang, X.—Lin, Z.: An Argumentation Framework for Description Logic Ontology Reasoning and Management. J. Intell. Inf. Syst., Vol. 40, 2013, No. 3, pp. 375–403, DOI:10.1007/s10844-012-0230-7.

[29] Ma, Y.—Hitzler, P.—Lin, Z.: Algorithms for Paraconsistent Reasoning with OWL. Proc. 4th European Semantic Web Conf., E. Franconi, M. Kifer and W. May (Eds.), Innsbruck 2007, pp. 399–413.

[30] Huang, Z.—van Harmelen, F.—ten Teije, A.: Reasoning with Inconsistent Ontologies. Proc. 19th Int. Joint Conf. Artif. Intell., L. Pack Kaelbling and A. Saffiotti (Eds.), Edinburgh 2005, pp. 454–459.

[31] Wang, Z.—Wang, K.—Topor, R. W.: A New Approach to Knowledge Base Revision in DL-Lite. Proc. 24th National Conf. Artif. Intell., M. Fox and D. Poole (Eds.), Atlanta 2010, pp. 369–374.

Xiaowang Zhang Ph. D., works in Databases and Theoretical Computer Science Group in Hasselt University, Belgium. His area of research is logics in computer science. He recent research interests include database querying languages, description logic ontology reasoning and management and multi-valued logic.

Zuoquan Lin Ph. D., works in School of Mathematical Sciences in Peking University, China. His area of research is Artificial Intelligence and intelligent software.