An argumentation workflow for reasoning in Ontology Based Data Access

Bruno Yun, Madalina Croitoru

An argumentation workflow for reasoning in Ontology Based Data Access

Bruno YUN [a], Madalina CROITORU [a,1]
[a] *INRIA Graphik/LIRMM University Montpellier, France*

Abstract. In this paper we demonstrate how to benefit from structured argumentation frameworks and their implementations to provide for reasoning capabilities of Ontology Based Data Access systems under inconsistency tolerant semantics. More precisely, given an inconsistent $Datalog^{\pm}$ knowledge base we instantiate it using the $ASPIC^{+}$ framework and show that the reasoning provided by $ASPIC^{+}$ is equivalent to the main inconsistent tolerant semantics in the literature. We provide a workflow that shows the practical interoperability of the logic based frameworks handling $Datalog^{\pm}$ and $ASPIC^{+}$.

Keywords. Applications and Structured Argumentation and Datalog+/-

Introduction and Motivation

Ontology Based Data Access (OBDA) is a popular setting used by many Semantic Web applications that encodes the *access to data sources using an ontology (vocabulary)* [17, 18,9]. The use of the ontology will help obtain a unified view over heterogeneous data sources. Moreover, the ontology will enable the exploitation of implicit knowledge not explicitly stored in the data sources alone. One of the main difficulties in OBDA consists in dealing with potentially inconsistent union of facts (data sources). Reasoning with inconsistency needs additional mechanisms because classical logic will infer everything out of *falsum*. It is classically assumed (and a hypothesis that we will also follow in this paper) that the inconsistency in OBDA occurs at the fact level and not due to the ontology [17,18,9]. The *facts are error prone* due to their unrestrained provenance while *ontologies are considered agreed upon* as shared conceptualisations.

We consider here two main methods of handling inconsistency. On one hand (and inspired from database research) we consider *repair based techniques*. A repair is a maximally consistent set of facts. Reasoning with inconsistency using repairs relies on reasoning with repairs and combining the results using various methods (called *inconsistency tolerant semantics*). [5,7,13] Despite them being the mainstream techniques for OBDA reasoning, the main drawback of inconsistent tolerant semantics is the *lack of implementations* excepting a few dedicated approaches to particular semantics [14,6].

A second method consists of using argumentation techniques. A Dung argumentation system [12] is a pair $\mathcal{AS} = (\mathcal{A}, \mathcal{C})$, where $\mathcal{A}$ is a set of arguments and $\mathcal{C} \subseteq \mathcal{A} \times \mathcal{A}$ is a binary attack relation on them.

[1]Corresponding Author: E-mail: croitoru@lirmm.fr

In this paper we demonstrate how to benefit from structured argumentation frameworks and their implementations to provide for reasoning capabilities of OBDA systems under inconsistency tolerant semantics. More precisely, given an inconsistent $Datalog^{\pm}$ [8] knowledge base we instantiate it using the $ASPIC^{+}$ framework [15] and show that the reasoning provided by $ASPIC^{+}$ is equivalent to the main inconsistent tolerant semantics in the literature. The *significance of this work* is a proposed workflow that will enable $Datalog^{\pm}$ frameworks to handle inconsistencies in knowledge bases by means of the $ASPIC^{+}$ framework. We use two frameworks:

- **Graal**, a Java toolkit dedicated to querying knowledge bases within the framework of $Datalog^{\pm}$ and maintained by GraphIK team. Graal takes as input a Dlgp file and a query and answer the query using various means (saturation, query rewriting). This toolkit can be found at `https://graphik-team.github.io/graal/`.
- **ACL's ASPIC** project that takes as input a query and $ASPIC^{+}$ knowledge base, i.e. rules (strict and defeasible), ordinary premises, axioms and preferences. The output is the answer to the query. This inference engine can be found at `http://aspic.cossac.org/components.html`.

We use Graal's representation of a knowledge base and construct the necessary input for the $ASPIC^{+}$ argumentation inference engine. The difficulty of this work resides in the definition of the mapping (the contrariness function and the way facts and rules are handled) that ensures the semantic equivalence proved in the next sections.

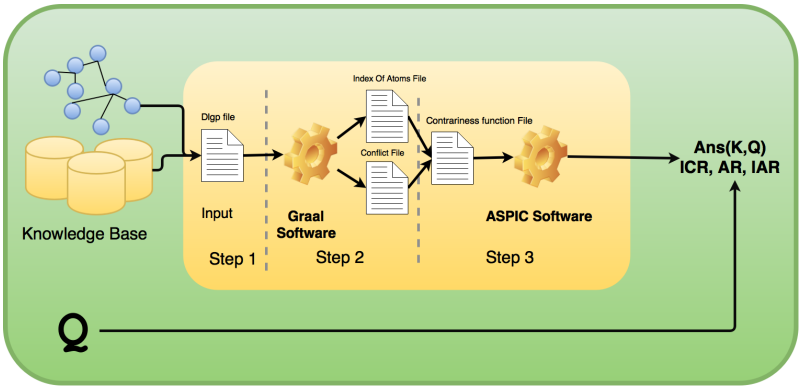

Figure 1. Interoperability Workflow of ACL and Graal.

In Figure 1 the interoperability workflow of the Graal software and the $ASPIC^{+}$ implementation are shown. Let us detail here how the workflow functions:

- **Step 1**. The input of the software is a $Datalog^{\pm}$ knowledge base obtained from the OBDA setting (that considers several data sources unified under the same ontology). The *dlgp* file that encodes this knowledge base (a textual format for the existential rule / Datalog framework) is parsed by the Graal framework. Each line in a *dlgp* file corresponds either to a fact, existential rule, negative constraint or conjunctive query. Please note that a complete grammar of the *dlgp* format is available here: `https://graphik-team.github.io/graal/papers/datalog+_v2.0_en.pdf`.

- **Step 2 and 3**. The intermediary files are meant to serve as input for the contrariness function computation in Step 3. The contrariness function encodes the conflicts between atoms. This will be formally defined in the next section. Please note that one of the main difficulties of this work is properly defining the contrariness function such that the produced results are sound and complete with respect to inconsistent tolerant semantics.
- **Querying**. The output of this inference engine is the answer to the query w.r.t the inconsistent knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$. The soundness and completeness of the answer with respect to inconsistency tolerant semantics is ensured by the equivalence results presented in the next section.

The Logical Language: $Datalog^{\pm}$

In this section we explain the logical language $Datalog^{\pm}$ used throughout the paper. We define the notion of $Datalog^{\pm}$ knowledge base, inconsistent knowledge base and explain the three inconsistency tolerant semantics mostly used in the literature. Its language $\mathcal{L}$ is composed of formulas built with the usual quantifier $(\exists, \forall)$ and *only* the connectors implication $(\rightarrow)$ and conjunction $(\wedge)$. We consider first-order vocabularies with constants but no other function symbol. An **atomic formula** (or atom) is of the form $p(t_1, ..., t_n)$ where $p \in \mathcal{P}$ is an n-ary predicate, and $t_1, ..., t_n$ are terms. Classically, a fact is a ground atom. We denote by $\vec{x}$ a vector of variables. An *existential rule* (or simply a rule) is a closed formula of the form $R = \forall\vec{x}\forall\vec{y}(B \rightarrow \exists\vec{z}H)$, where B and H are conjuncts, with *vars*$(B) = \vec{x} \cup \vec{y}$, and *vars*$(H) = \vec{x} \cup \vec{z}$. The variables $\vec{z}$ are called the existential variables of the rule R. B and H are respectively called the *body* and the *head* of R. We denote them respectively $body(R)$ for B and $head(R)$ for H. We may sometimes omit quantifiers and write $R = B \rightarrow H$. A *negative constraint* (or simply a constraint) is a rule of the form $N = \forall\vec{x}(B \rightarrow \bot)$. A rule $R = B \rightarrow H$ is **applicable** to a fact F if there is a homomorphism σ from B to F. Let F be a fact and $\mathcal{R}$ be a set of rules. A fact F' is called an $\mathcal{R}$**-derivation** of F if there is a finite sequence (called the **derivation sequence**) $(F_0 = F, ..., F_n = F')$ such that for all $0 \leq i < n$ there is a rule R which is applicable to F_i and F_{i+1} is an immediate derivation from F_i. Given a fact F and a set of rules $\mathcal{R}$, the **chase** (or saturation) procedure starts from F and performs rule applications in a breadth-first manner. The chase computes the **closure** of F, i.e. $CL_{\mathcal{R}}(F)$, which is the smallest set that contains F and that is closed under R-derivation, i.e. for every $\mathcal{R}$-derivation F' of F we have $F' \in CL_{\mathcal{R}}(F)$. Given a chase variant C [4], we call C-finite the class of set of rules $\mathcal{R}$, such that the C-chase halts on any fact F, consequently produces a finite $CL_{\mathcal{R}}(F)$. We limit our work in this paper to these kind of classes.

Let F and F' be two facts. $F \models F'$ if and only if there is a homomorphism from F' to F. Given two facts F and F' and a set of rules $\mathcal{R}$ we say $F, \mathcal{R} \models F'$ if and only if $CL_{\mathcal{R}}(F) \models F'$ where $\models$ is the classical first-order entailment [16].

Knowledge base and inconsistency Let us denote by $\mathcal{L}$ the language described so far, A knowledge base $\mathcal{K}$ is a finite subset of $\mathcal{L}$. Precisely, $\mathcal{K}$ is a tuple $(\mathcal{F}, \mathcal{R}, \mathcal{N})$ of a finite set of facts $\mathcal{F}$, rules $\mathcal{R}$ and constrains $\mathcal{N}$. Saying that $\mathcal{K} \models F$ means $CL_{\mathcal{R}}(\mathcal{F}) \models F$. We say a set of facts $\mathcal{F}$ is $\mathcal{R}$-inconsistent with respect to a set of constraints $\mathcal{N}$ and rules $\mathcal{R}$ if and only if there exists $N \in \mathcal{N}$ such that $CL_{\mathcal{R}}(\mathcal{F}) \models body(N)$, otherwise $\mathcal{F}$ is $\mathcal{R}$-consistent. A knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is said to be inconsistent with respect

to $\mathcal{R}$ and $\mathcal{N}$ (inconsistent for short) if $\mathcal{F}$ is $\mathcal{R}$-inconsistent. We may use the notation $CL_{\mathcal{R}}(\mathcal{F}) \models \bot$ to mean the same thing.

In the area of inconsistent ontological knowledge base query answering, we usually check what can be inferred from an inconsistent ontology. We usually begin by calculating all maximal consistent subsets of $\mathcal{K}$ called *repairs*. Given a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$, we call by $Repairs(\mathcal{K})$ the set of all repairs defined as:

$$Repairs(\mathcal{K}) = \{\mathcal{F}' \subseteq \mathcal{F} | \mathcal{F}' \text{ is maximal for } \subseteq \text{ and } \mathcal{R}\text{-consistent}\}$$

Different inconsistency tolerant semantics are used for inconsistent ontology knowledge base query answering (Intersection of All Repairs: IAR, All Repairs: AR, Intersection of Closed Repairs: ICR); these semantics can yield different results.

Definition 1 ***[cf [11]].*** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and α be a query.*

- *α is* AR-entailed *from $\mathcal{K}$: $\mathcal{K} \models_{AR} \alpha$ iff for $\forall r \in Repairs(\mathcal{K}), Cl_{\mathcal{R}}(r) \models \alpha$.*
- *α is* ICR-entailed *from $\mathcal{K}$, written $\mathcal{K} \models_{ICR} \alpha$ iff $\bigcap_{r \in Repairs(\mathcal{K})} Cl_{\mathcal{R}}(r) \models \alpha$*
- *α is* IAR-entailed *from $\mathcal{K}$, written $\mathcal{K} \models_{IAR} \alpha$ iff $CL_{\mathcal{R}}(\bigcap_{r \in Repairs(\mathcal{K})} r) \models \alpha$*

Structured Argumentation for $Datalog^{\pm}$

In this section we address the problem of how to use structured argumentation for $Datalog^{\pm}$. We show how the $ASPIC^{+}$ framework can be instantiated to yield results equivalent to the state of the art in OBDA inconsistency tolerant semantics. We define the first instantiation in the literature of $ASPIC^{+}$ using $Datalog^{\pm}$.

$ASPIC^{+}$ [15] is a framework for obtaining logical based argumentation system using any logical language $\mathcal{L}$. It is meant to generate an abstract argumentation framework and was created because abstract argumentation does not specify the structure of arguments and the nature of attacks. $ASPIC^{+}$ is meant to provide guidance to those aspects without losing a large range of instantiating logics. Before going any further we will provide a few basic abstract argumentation notions needed later in this section. We consider $\mathcal{AS} = (\mathcal{A}, \mathcal{C})$, where $\mathcal{A}$ is a set of arguments and $\mathcal{C} \subseteq \mathcal{A} \times \mathcal{A}$ is a binary attack relation on them. We say that the argument $a \in \mathcal{A}$ is *acceptable* w.r.t a set of arguments $\varepsilon \subseteq \mathcal{A}$ iff $\forall b \in \mathcal{A}$ such that $(b, a) \in \mathcal{C}, \exists c \in \varepsilon$ such that $(c, b) \in \mathcal{C}$. ε is *conflict-free* iff $\nexists a, b \in \varepsilon$ such that $(a, b) \in \mathcal{C}$. ε is *admissible* iff ε is conflict-free and all arguments of ε are acceptable w.r.t ε. ε is preferred iff it is maximal (for set inclusion) and admissible. ε is *stable* iff it is conflict-free and $\forall a \in \mathcal{A} \backslash \varepsilon, \exists b \in \varepsilon$ such that $(b, a) \in \mathcal{C}$. ε is *complete* iff it contains all arguments that are acceptable w.r.t ε. ε is *grounded* iff it is minimal (for set inclusion) and complete. Reasoning takes place on the various ε (also called extensions). Following [15], to use $ASPIC^{+}$, we need to choose a logical language $\mathcal{L}$ closed under negation ($\neg$), provide a set of rules $\mathcal{R} = \mathcal{R}_d \cup \mathcal{R}_s$ composed of defeasible rules and strict rules with $\mathcal{R}_d \cap \mathcal{R}_s = \emptyset$, specify a contrariness function $cf : \mathcal{L} \to 2^{\mathcal{L}}$ and a partial naming function $n : \mathcal{R}_d \to \mathcal{L}$ that associates a well-formed formulas of $\mathcal{L}$ to a defeasible rule. The function n will not be used in this instantiation. In $ASPIC^{+}$ an argumentation system is a triple $\mathcal{AS} = (\mathcal{L}, \mathcal{R}, n)$ and a knowledge base is $\mathcal{K} \subseteq \mathcal{L}$ consisting of two disjoint subsets $\mathcal{K}_n$ (the axioms) and $\mathcal{K}_p$ (the ordinary premises).

To instantiate $ASPIC^+$ for $Datalog^{\pm}$, we define $\mathcal{L}$ as $Datalog^{\pm}$, rules in definition 2 and the contrariness function in definition 3. Please note that definition 4 and 7 are new w.r.t. state of art regarding $Datalog^{\pm}$ instantiations conform [15].

Definition 2 *Strict rules (resp. defeasible rules) are of the form* $\forall\vec{x}\forall\vec{y}(B \rightarrow \exists\vec{z}H)$ *(resp.* $\forall\vec{x}\forall\vec{y}(B \Rightarrow \exists\vec{z}H)$ *) with* B*, the body and* H*, the head are atoms or conjunction of atoms with* $\text{vars}(B) = \vec{x} \cup \vec{y}$*, and* $\text{vars}(H) = \vec{x} \cup \vec{z}$*..*

Definition 3 ***[[15]].*** *The function* cf *is a function from* $\mathcal{L}$ *to* $2^{\mathcal{L}}$ *such that:*

- φ *is the contrary of* ψ *if* $\varphi \in cf(\psi), \psi \notin cf(\varphi)$
- φ *is the contradictory of* ψ *if* $\varphi \in cf(\psi), \psi \in cf(\varphi)$
- *Each* $\varphi \in \mathcal{L}$ *has at least one contradictory.*

We define our own contrariness function to instantiate $ASPIC^+$ for $Datalog^{\pm}$ ($\mathcal{L} = Datalog^{\pm}$). This contrariness function is necessary because it is used in the attack relation. It is worth noting that the idea that we want to capture (as also defined in [1]) is that x is the contrary of y iff they cannot be both true but they can be both false. They are contradictory if the truth of one implies the falsity of the other and vise versa.

Definition 4 ($Datalog^{\pm}$ **contrariness function)** *Let* $a \in \mathcal{L}$ *and* b *be a conjunction of atoms.* $b \in cf(a)$ *iff* $\exists\psi$ *an atom such that* $a \models \psi$ *and* $\{b, \psi\}$ *is* $\mathcal{R}$*-inconsistent.*

Here we recall that an $ASPIC^+$ argument can be built from axioms and ordinary premises or from rules and other arguments. The arguments are built once $\mathcal{R}_d, \mathcal{R}_s, cf$ and $\mathcal{K}$ are known.

Definition 5 (Argument cf [15]) *Arguments in* $ASPIC^+$ *can be in two forms:*

- $\emptyset \rightarrow c$ *(resp.* $\emptyset \Rightarrow c$*) with* $c \in \mathcal{K}_n$ *(resp.* $c \in \mathcal{K}_p$ *or* $\emptyset \Rightarrow c \in \mathcal{R}_d$*) such that* $Prem(A) = \{c\}, Conc(A) = c,$ *Sub*$(A) = \{A\}$ *with* $Prem$ *returns premisses of* A *and* $Conc$ *returns its conclusion.*
  $DefRules(A) = \emptyset$*.*
- $A_1, ..., A_m \rightarrow c$ *(resp.* $A_1, ..., A_m \Rightarrow c$*), such that there exists a strict (resp. defeasible) rule* $r = B \rightarrow H$ *(resp.* $r = B \Rightarrow H$*) and a homomorphism* σ *from* B *to* $X = Conc(A_1) \wedge Conc(A_2) \wedge \cdots \wedge Conc(A_m)$*.*
  $Prem(A) = Prem(A_1) \cup ... \cup Prem(A_m)$*,*
  $Conc(A) = c = \alpha(X, r, \sigma)$*,*
  Sub$(A) =$ *Sub*$(A_1) \cup ... \cup$ *Sub*$(A_m) \cup \{A\}$*,*
  $TopRule(A) =$ *rule* $r = B \rightarrow H$ *(resp.* $r = B \Rightarrow H$*), such that there exists an homomorphism* σ *from* B *to* X*.*
  $DefRules(A) = DefRules(A_1) \cup \cdots \cup DefRules(A_m)$ *(resp.* $DefRules(A) = DefRules(A_1) \cup \cdots \cup DefRules(A_m) \cup \{TopRule(A)\}$*).*

Attacks in $ASPIC^+$ are based on three notions (undercutting, undermining and rebutting). Each of those notions are useful as they capture different aspects of conflicts. In short, arguments can be attacked on a conclusion of a defeasible inference (rebutting attack), on a defeasible inference step itself (undercutting attack), or on an ordinary premise (undermining attack).

Definition 6 ***[cf [15]].*** *Let a and b be arguments, we say that a attacks b iff a undercuts, undermines or rebuts b, where:*

- *a undercuts argument b (on b') iff $Conc(a) \in cf(n(r))$ for some $b' \in Sub(b)$ such that b''s top rule r is defeasible.*
- *a rebuts argument b (on b') iff $Conc(a) \in cf(\psi)$ for some $b' \in Sub(b)$ of the form $b'_0, \ldots, b'_n \Rightarrow \psi$.*
- *a undermines b (on ψ) iff $Conc(a) \in cf(\psi)$ for an ordinary premise ψ of b.*

We are now ready to define the mapping that allows the instantiation of $ASPIC^+$ with $Datalog^{\pm}$. The mapping will consider each fact as a defeasible rule because the inconsistency in the OBDA setting is assumed to come from the facts level. Therefore the only attack we consider in this instantiation is the undermine attack because we have simple defeasible rules. The rules of the ontology become strict rules.

Definition 7 (Mapping For $ASPIC^+$ Instantiation of $Datalog^{\pm}$) *We denote by $\mathcal{S}$ the set of all possible inconsistent knowledge bases of the form $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ and $\mathcal{G}$ the set of all $ASPIC^+$ instantiation using $Datalog^{\pm}$ language. The mapping $\tau : S \rightarrow \mathcal{G}$ is defined as follows:*

1. *The mapping τ associates every $\mathcal{R}$-consistent subset $F_i \subseteq \mathcal{F}$ to its defeasible rule $\emptyset \Rightarrow conjunct(F_i)$ where $conjunct(F_i)$ denotes the conjunction of facts contained in F_i.*
2. *The mapping τ associates every rules $r_i \in \mathcal{R}$ to the same rule $r_i \in \mathcal{R}_s$.*

We will considerate that if $\emptyset \Rightarrow c$, then c is an ordinary premise $(c \in \mathcal{K}_p)$.

In order to give properties of the $ASPIC^+$ instantiation presented in this paper we remind few notions. ε is *admissible* iff ε is conflict-free and all arguments of ε are acceptable w.r.t ε. ε is preferred iff it is maximal (for set inclusion) and admissible. ε is *stable* iff it is conflict-free and $\forall a \in \mathcal{A} \backslash \varepsilon, \exists b \in \varepsilon$ such that $(b, a) \in \mathcal{C}$.

We denote by $AF^A_{\mathcal{K}}$ the $ASPIC^+$ argumentation framework constructed from $\mathcal{K}$ using the mapping of definition 7. We restate that attacks in $AF^A_{\mathcal{K}}$ are composed only of undermining because we only have simple defeasible rules of the form $\emptyset \Rightarrow c$. The following lemma shows that stable extensions are closed under sub-arguments in the $Datalog^{\pm}$ instantiation of $ASPIC^+$.

Lemma 1 *Let ε be an $ASPIC^+$ stable extension and $A \in \varepsilon$ an argument contained in ε. Then $Sub(A) \subseteq \varepsilon$.*

Notation Let $c = \alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_n$ be a conjunction of facts. $Elimination(c) = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ is the set resulting from eliminating the conjunction of c. Let S be a set of facts. We denote by $\mathcal{P}(S)$ the superset of S which correspond to all subsets of S.

We can now define the set of arguments constructed on a consistent set of facts.

Definition 8 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and $AF^A_{\mathcal{K}}$ be the corresponding $ASPIC^+$ instantiation and $S \subseteq \mathcal{F}$ a $\mathcal{R}$-consistent subset of $\mathcal{F}$. We denote by $Arg^A(S)$ the set of arguments such that their premises are contained in S. Formally:*

$$Arg^A(S) = \{ASPIC^+ \textit{ argument } a | \bigcup_{c \in Prem(a)} Elimination(c) \subseteq \mathcal{P}(S)\}$$

The main result shows that the set of stable extension coincides with the set of preferred one and it is obtained from the arguments built on repairs.

Theorem 1 (Repair Equivalence for $ASPIC^+$ Instantiation) *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $AF^A_{\mathcal{K}}$ be the corresponding $ASPIC^+$ instantiation and $\sigma \in \{preferred, stable\}$. Then:*

$$\{Arg^A(R) | R \in Repair(\mathcal{K})\} = Ext_\sigma(AF^A_{\mathcal{K}})$$

The state of the art can also provide a structured argumentation framework of $Datalog^{\pm}$ [11,10]. Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, we denote by $AF^M_{\mathcal{K}}$ the instantiated logical argumentation framework $(\mathcal{A}, \mathcal{C})$ with $\mathcal{A} = Arg(\mathcal{F})$ and $\mathcal{C}$ defined in [11,10]. According to [11] the arguments constructed on the set of repairs coincide with the arguments in the stable and preferred extension: $\{Arg(R) | R \in Repair(\mathcal{K})\} = Ext_\sigma(AF^M_{\mathcal{K}})$. We can thus conclude that the preferred/stable extensions in the two instantiated frameworks are the same and that for each stable/preferred extension of one framework, there is a corresponding stable/preferred extension in the other and vice-versa. This is formalized in the theorem below.

Theorem 2 (Instantiations Equivalence) *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $AF^M_{\mathcal{K}}$ and $AF^A_{\mathcal{K}}$ be the two argumentation framework instantiations. Then if $\sigma \in \{preferred, stable\}$, $|Ext_\sigma(AF^M_{\mathcal{K}})| = |Ext_\sigma(AF^A_{\mathcal{K}})|$ and for each extension under semantics $\sigma, \varepsilon \in Ext_\sigma(AF^M_{\mathcal{K}})$, there is a corresponding extension $\varepsilon_2 \in Ext_\sigma(AF^A_{\mathcal{K}})$ and vice-versa (the corresponding extension can be found via repairs).*

Conclusions

In this paper we demonstrated how to benefit from structured argumentation frameworks and their implementations to provide for reasoning capabilities of OBDA systems under inconsistency tolerant semantics. More precisely, given an inconsistent $Datalog^{\pm}$ knowledge base we instantiated it using the $ASPIC^+$ framework and showed that the reasoning provided by $ASPIC^+$ is equivalent to the main inconsistent tolerant semantics in the literature. A workflow of interoperability between $ASPIC^+$ ACL framework and Graal $Datalog^{\pm}$ framework was thus formally underpinned. In future work we are interested in exploiting this workflow for the explanation capabilities of inconsistent tolerant semantics [2,3].

Acknowledgments

The authors acknowledge the support of ANR grant QUALINCA (ANR-12-0012).

References

[1] L. Amgoud. Five weaknesses of ASPIC +. In *Advances in Computational Intelligence - 14th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2012, Catania, Italy, July 9-13, 2012, Proceedings, Part III*, pages 122–131, 2012.

[2] A. Arioua and M. Croitoru. A dialectical proof theory for universal acceptance in coherent logic-based argumentation frameworks. *Proc of ECAI 2016 - to appear.*

[3] A. Arioua and M. Croitoru. Dialectical characterization of consistent query explanation with existential rules. In *FLAIRS: Florida Artificial Intelligence Research Society*, 2016.

[4] J.-F. Baget, F. Garreau, M.-L. Mugnier, and S. Rocher. Revisiting Chase Termination for Existential Rules and their Extension to Nonmonotonic Negation. *ArXiv e-prints*, May 2014.

[5] M. Bienvenu. On the complexity of consistent query answering in the presence of simple ontologies. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*, 2012.

[6] M. Bienvenu, C. Bourgaux, and F. Goasdoué. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *AAAI*, pages 996–1002, 2014.

[7] M. Bienvenu and R. Rosati. Tractable approximations of consistent query answering for robust ontology-based data access. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, 2013.

[8] A. Calì, G. Gottlob, and T. Lukasiewicz. Datalog$+/-$: a unified approach to ontologies and integrity constraints. In *Proc of ICDT'09*, pages 14–30. ACM, 2009.

[9] A. Calì, G. Gottlob, and T. Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. *J. Web Sem.*, 14:57–83, 2012.

[10] M. Croitoru, R. Thomopoulos, and S. Vesic. Introducing preference-based argumentation to inconsistent ontological knowledge bases. In *PRIMA 2015: Principles and Practice of Multi-Agent Systems - 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings*, pages 594–602, 2015.

[11] M. Croitoru and S. Vesic. What can argumentation do for inconsistent ontology query answering? In *Scalable Uncertainty Management - 7th International Conference, SUM 2013, Washington, DC, USA, September 16-18, 2013. Proceedings*, pages 15–29, 2013.

[12] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[13] D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, and D. F. Savo. Inconsistency-tolerant semantics for description logics. In *Web Reasoning and Rule Systems - Fourth International Conference, RR 2010, Bressanone/Brixen, Italy, September 22-24, 2010. Proceedings*, pages 103–117, 2010.

[14] M. V. Martinez, C. A. D. Deagustini, M. A. Falappa, and G. R. Simari. Inconsistency-tolerant reasoning in datalogˆ{\ pm} ontologies via an argumentative semantics. In *Ibero-American Conference on Artificial Intelligence*, pages 15–27. Springer, 2014.

[15] S. Modgil and H. Prakken. The *ASPIC*$^+$ framework for structured argumentation: a tutorial. *Argument & Computation*, 5(1):31–62, 2014.

[16] M. Mugnier. Ontological query answering with existential rules. In *Web Reasoning and Rule Systems - 5th International Conference, RR 2011, Galway, Ireland, August 29-30, 2011. Proceedings*, pages 2–23, 2011.

[17] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. Data Semantics*, 10:133–173, 2008.

[18] L. Popa, S. Abiteboul, and P. G. Kolaitis, editors. *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA*. ACM, 2002.