A Short Survey on Inconsistency Handling in Ontology-Mediated Query Answering

Meghyn Bienvenu

Noname manuscript No.
(will be inserted by the editor)

A Short Survey on Inconsistency Handling in Ontology-Mediated Query Answering

Meghyn Bienvenu

Received: date / Accepted: date

Abstract This paper provides a concise overview of the literature on inconsistency handling for ontology-mediated query answering, a topic which has grown into an active area of research over the last decade. The focus of this survey is on the case where errors are localized in the data (i.e., the ontology is deemed reliable) and where inconsistency-tolerant semantics are employed with the aim of obtaining meaningful information from inconsistent knowledge bases.

Keywords Inconsistency Handling · Ontology-mediated Query Answering · Description Logics

1 Introduction

It is widely acknowledged that real-world data is plagued by numerous data quality issues, among them the presence of erroneous facts. While already a serious issue for 'plain' databases, the problem of handling imperfect data is even more critical in the setting of ontology-mediated query answering (OMQA), where an ontology is used to enrich the data with domain knowledge. Indeed, even a single erroneous fact can provoke a logical inconsistency, thereby rendering classical OMQA semantics (based upon first-order logic) useless, since everything is entailed from a contradiction. This has motivated researchers from knowledge representation and reasoning, and especially those from the description logic (DL) community, to study a variety of approaches for handling inconsistent data in OMQA, adapting and extending techniques initially proposed for databases. Now that there has been over a decade of research on inconsistency handling in OMQA, the time is ripe to take a step back and evaluate the progress that has been made and what remains to be done.

In this paper, we will try to summarize what is now quite a large body of work related to inconsistency handling in OMQA. Our treatment will necessarily be incomplete. We will concentrate on the case in which inconsistencies are due to errors in the data (i.e., we assume the ontology has been properly debugged) and mainly discuss how inconsistency-tolerant semantics can be used to obtain meaningful information from inconsistent knowledge bases. While our focus will be on ontologies formulated using DLs, the inconsistency-tolerant semantics presented in this chapter are language-agnostic and can be applied to any ontology language. In particular, there have been several works (see e.g. [41,40,5,39]) which have explored such semantics for existential rules (aka Datalog +/-) [20, 42], which constitute another prominent class of ontology languages. Our treatment is based upon (and complementary to) a much more detailed tutorial chapter [12] and incorporates some more recent literature and perspectives for future work.

2 Preliminaries

We briefly recall here some useful DL terminology and notation, and we direct readers to [4] for a comprehensive introduction to DLs. Throughout the paper, we shall assume that $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is a DL *knowledge base* (KB) composed of a *TBox* $\mathcal{T}$ and *ABox* $\mathcal{A}$. As usual, the TBox $\mathcal{T}$ will be a finite set of axioms whose syntax is dictated by the chosen DL, while the ABox $\mathcal{A}$

M. Bienvenu
LaBRI – CNRS & Université de Bordeaux
351 Cours de la Libération, F-33405 Talence cedex, France
Tel.: +33-(0)5 40 00 38 72
Fax: +33-(0)5 40 00 66 69
E-mail: meghyn.bienvenu@labri.fr

is a finite set of assertions (ground facts). Our running example (borrowed from [12]) uses the following KB formulated in the core dialect of DL-Lite, a popular family of lightweight DLs [22]:

Example 1 Consider the TBox $\mathcal{T}_{\mathsf{univ}}$ with the axioms:

$$\begin{array}{lll} \mathsf{Prof} \sqsubseteq \mathsf{Faculty} & \mathsf{Prof} \sqsubseteq \exists \mathsf{Teaches} & \mathsf{Prof} \sqsubseteq \neg \mathsf{Lect} \\ \mathsf{Lect} \sqsubseteq \mathsf{Faculty} & \mathsf{Lect} \sqsubseteq \exists \mathsf{Teaches} & \mathsf{Prof} \sqsubseteq \neg \mathsf{Fellow} \\ \mathsf{Fellow} \sqsubseteq \mathsf{Faculty} & \exists \mathsf{Teaches}^- \sqsubseteq \mathsf{Course} & \mathsf{Lect} \sqsubseteq \neg \mathsf{Fellow} \\ \mathsf{Faculty} \sqsubseteq \neg \mathsf{Course} & & \end{array}$$

where Prof, Lect, Fellow, Faculty, and Course are concept names (unary predicates) that represent the classes of professors, lecturers, research fellows, faculty members, and courses, respectively, and Teaches is a role (binary relation) linking teachers to what is taught. The TBox axioms state collectively that professors, lecturers, and fellows are three pairwise-disjoint classes of faculty, that professors and lecturers must teach something (i.e. occur in the first argument of some Teaches fact), that faculty and courses are disjoint, and that the second argument of Teaches ranges over courses.

The ABox $\mathcal{A}_{\mathsf{univ}}$ contains assertions about specific people and courses:

$$\begin{aligned} \mathcal{A}_{\mathsf{univ}} = \{&\mathsf{Prof}(\mathsf{sam}), \mathsf{Lect}(\mathsf{sam}), \mathsf{Fellow}(\mathsf{sam}), \mathsf{Prof}(\mathsf{kim}), \\ &\mathsf{Lect}(\mathsf{kim}), \mathsf{Fellow}(\mathsf{jane}), \mathsf{Fellow}(\mathsf{alex}), \\ &\mathsf{Teaches}(\mathsf{cs34}, \mathsf{jane}), \mathsf{Teaches}(\mathsf{alex}, \mathsf{cs48})\} \end{aligned}$$

Here for example, the assertion Prof(sam) state that sam is a professor, while Teaches(alex, cs48) states that alex teaches cs48.

Every DL KB can be translated into a first-order logic formula, and DL semantics corresponds to classical first-order semantics, in which interpretations give meaning to the basic symbols. We denote by $\mathcal{I} \models \alpha$ that the interpretation $\mathcal{I}$ satisifies the (ABox or Tbox) statement α. An interpretation $\mathcal{I}$ that satisfies all statements of the KB $\langle \mathcal{T}, \mathcal{A} \rangle$ is called a *model* of $\langle \mathcal{T}, \mathcal{A} \rangle$, and a KB is said to be *consistent* (or satisfiable) if has at least one model. An ABox $\mathcal{A}$ is *$\mathcal{T}$-consistent* if the KB $\langle \mathcal{T}, \mathcal{A} \rangle$ is consistent, and otherwise, it is $\mathcal{T}$-inconsistent.

We recall that a *conjunctive query* (CQ) takes the form of a conjunction of relational atoms where some of the variables occurring in atoms may be existentially quantified. In the DL setting, the relations occurring in atoms will be either concept or role names. An *instance query* (IQ) is a CQ which has a single atom and no existentially quantified variables. The *arity* of a CQ is the number of its free variables. Under classical OMQA semantics, we are interested in finding *certain answers* of a CQ q w.r.t. a KB $\langle \mathcal{T}, \mathcal{A} \rangle$, i.e. those tuples $\mathbf{a}$ of constants from $\mathcal{A}$ of the same arity as q such that $q(\mathbf{a})$ (i.e. the first-order sentence obtained by substituting $\mathbf{a}$ for the free variables of q) holds in every model of $\langle \mathcal{T}, \mathcal{A} \rangle$. The notation $\mathcal{K} \models q(\mathbf{a})$ indicates that $\mathbf{a}$ is a certain answer to q over $\mathcal{K}$. We call a subset $C \subseteq \mathcal{A}$ a *$\mathcal{T}$-support* of $q(\mathbf{a})$ if C is $\mathcal{T}$-consistent and $\langle \mathcal{T}, C \rangle \models q(\mathbf{a})$.

3 Inconsistency-Tolerant Semantics

As mentioned in the introduction, the usual first-order semantics of DLs does not provide any useful information when the KB is inconsistent, as everything can be inferred from a contradiction. To address this limitation, several inconsistency-tolerant semantics have been proposed with the aim of returning meaningful answers to queries posed over inconsistent KBs.

A key notion that underlies many of the proposed semantics is that of a repair, which intuitively captures the different ways of restoring consistency while retaining as much of the original information as possible. If we use set inclusion to select the maximal ABoxes, as was proposed in [32] and many subsequent works, then repairs can be formalized as follows.

Definition 1 An *(ABox) repair* of an ABox $\mathcal{A}$ w.r.t. a TBox $\mathcal{T}$ is an inclusion-maximal subset of $\mathcal{A}$ that is $\mathcal{T}$-consistent. We use $Rep(\mathcal{A}, \mathcal{T})$ to denote the set of repairs of $\mathcal{A}$ w.r.t. $\mathcal{T}$, which we abbreviate to $Rep(\mathcal{K})$ when $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$.

Example 2 The KB $\langle \mathcal{T}_{\mathsf{univ}}, \mathcal{A}_{\mathsf{univ}} \rangle$ is inconsistent and has 12 repairs. For example, it is easily verified that the following two ABoxes are both repairs:

$$\begin{aligned} \mathcal{R}_1 = \{&\mathsf{Prof}(\mathsf{sam}), \mathsf{Prof}(\mathsf{kim}), \mathsf{Fellow}(\mathsf{jane}), \mathsf{Fellow}(\mathsf{alex}), \\ &\mathsf{Teaches}(\mathsf{alex}, \mathsf{cs48})\} \\ \mathcal{R}_2 = \{&\mathsf{Fellow}(\mathsf{sam}), \mathsf{Lect}(\mathsf{kim}), \mathsf{Teaches}(\mathsf{cs34}, \mathsf{jane}), \\ &\mathsf{Fellow}(\mathsf{alex}), \mathsf{Teaches}(\mathsf{alex}, \mathsf{cs48})\} \end{aligned}$$

Each repair is $\mathcal{T}$-consistent, so it is possible to query a repair using classical semantics. The difficulty, however, is that there are typically several different repairs of an inconsistent KB, so we need to decide how to combine the answers obtained from the different repairs. Arguably the most natural approach is to require that a tuple be a certain answer no matter which repair is considered. This idea is captured by the AR semantics, which was first defined in [32] and can be seen as the OMQA analog of the consistent query answering approach long studied in the database literature [1,24,8].

Definition 2 (AR semantics) A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ under the *AR (ABox Repair) semantics*, written $\mathcal{K} \models_{\mathsf{AR}} q(\mathbf{a})$, just in the case that $\langle \mathcal{T}, \mathcal{B} \rangle \models q(\mathbf{a})$ for every repair $\mathcal{B} \in Rep(\mathcal{K})$.

A more conservative semantics, termed the IAR semantics [32], is obtained by querying the intersection of the repairs (or equivalently, the set of assertions not participating in any minimal inconsistent subset).

Definition 3 (IAR semantics) A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K}$ under the *IAR (Intersection of ABox Repairs) semantics*, written $\mathcal{K} \models_{\mathsf{IAR}} q(\mathbf{a})$, just in the case that $\langle \mathcal{T}, \mathcal{D} \rangle \models q(\mathbf{a})$ where $\mathcal{D} = \bigcap_{\mathcal{B} \in Rep(\mathcal{K})} \mathcal{B}$.

The more adventurous brave semantics, first explored in the OMQA setting in [18], merely requires that an answer hold w.r.t. at least some repair.

Definition 4 (Brave semantics) A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ under the *brave semantics*, written $\mathcal{K} \models_{\mathsf{brave}} q(\mathbf{a})$, just in the case that $\langle \mathcal{T}, \mathcal{B} \rangle \models q(\mathbf{a})$ for some repair $\mathcal{B} \in Rep(\mathcal{K})$.

Before proceeding further, let us illustrate the AR, IAR, and brave semantics on our running example:

Example 3 If we evaluate the query $q(x) = \mathsf{Faculty}(x)$ using the three semantics, we obtain:

- 3 answers for AR semantics: sam, kim, alex
- 1 answer for IAR semantics: alex
- 4 answers for brave semantics: sam, kim, alex, jane

The preceding three semantics are related as follows:

$$\mathcal{K} \models_{\mathsf{IAR}} q(\mathbf{a}) \quad \Rightarrow \quad \mathcal{K} \models_{\mathsf{AR}} q(\mathbf{a}) \quad \Rightarrow \quad \mathcal{K} \models_{\mathsf{brave}} q(\mathbf{a})$$

In other words, the brave and IAR semantics provide respectively upper and lower bounds on the set of answers w.r.t. the AR semantics.

We can also compare semantics based upon the properties they satisfy. Following [12], we consider the following three desirable properties for an inconsistency-tolerant semantics:

Consistent Support Semantics S has the Consistent Support property if for every KB $\langle \mathcal{T}, \mathcal{A} \rangle$, query q, and tuple $\mathbf{a}$, if $\langle \mathcal{T}, \mathcal{A} \rangle \models_S q(\mathbf{a})$, then there exists a $\mathcal{T}$-support $C \subseteq \mathcal{A}$ of $q(\mathbf{a})$.

Consistent Results Semantics S has the Consistent Results property if for every KB $\langle \mathcal{T}, \mathcal{A} \rangle$, there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $\mathcal{I} \models q(\mathbf{a})$ for every $q(\mathbf{a})$ with $\langle \mathcal{T}, \mathcal{A} \rangle \models_S q(\mathbf{a})$.

Unique Base Semantics S has the Unique Base property if for every KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, there exists a $\mathcal{T}$-consistent ABox $\mathcal{A}'$ such that for every query q and tuple $\mathbf{a}$: $\langle \mathcal{T}, \mathcal{A} \rangle \models_S q(\mathbf{a})$ iff $\langle \mathcal{T}, \mathcal{A}' \rangle \models q(\mathbf{a})$.

The Consistent Support property means that every answer can be justified by exhibiting a consistent subset of the original ABox. The interest of Consistent Results is that it allows users to safely combine the query results obtained under semantics S (in the sense that no contradiction can be inferred from the returned information). Finally, the Unique Base property is a nice feature from the implementation point of view, since it means we can compute in an offline phase a consistent ABox, which can be queried using existing algorithms.

	Semantics with the property
Consistent Results	non-objection, CAR, ICAR, AR, ICR, k-support, k-lazy, IAR
Consistent Support	brave, k-defeater, non-objection, AR, ICR,k-support, k-lazy, IAR
Unique Base	IAR, ICR, ICAR

Fig. 1: Properties of inconsistency-tolerant semantics.

As seen in Fig. 1, the brave semantics satisfies only Consistent Support, the AR semantics satisfies both Consistent Support and Consistent Results, while the IAR semantics satisfies all three properties.

Let us now continue on to other semantics that have been proposed in the OMQA literature, starting with the ICR semantics, defined in [10]:

Definition 5 (ICR semantics) Let $\mathsf{close}_{\mathcal{T}}(\mathcal{B})$ contain all ABox assertions β such that $\langle \mathcal{T}, \mathcal{B} \rangle \models \beta$. A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ under the *ICR (Intersection of Closed Repairs) semantics* just in the case that $\langle \mathcal{T}, \mathcal{D} \rangle \models q(\mathbf{a})$ where $\mathcal{D} = \bigcap_{\mathcal{B} \in Rep(\mathcal{K})} \mathsf{close}_{\mathcal{T}}(\mathcal{B})$.

By closing repairs before intersecting them, the ICR semantics provides a better lower approximation of the AR semantics than the IAR semantics, and it can be shown to satisfy the three properties. This semantics coincides with the AR semantics on instance queries, which means that for our example KB, the ICR semantics would return sam, kim, alex as answers to the query $q(x) = \mathsf{Faculty}(x)$.

The idea of adding inferred assertions to retain more information is also at the heart of the CAR and ICAR semantics proposed in [32]. The key difference is that a modified closure operator is applied to the original inconsistent ABox, and the enriched ABox is then used to define closed ABox repairs.

Definition 6 (Closed ABox repair) Let $\mathsf{close}^*_{\mathcal{T}}(\mathcal{A})$ contain all ABox assertions β such that there is a $\mathcal{T}$-consistent subset $S \subseteq \mathcal{A}$ such that $\langle \mathcal{T}, S \rangle \models \beta$. A subset $\mathcal{R} \subseteq \mathsf{close}^*_{\mathcal{T}}(\mathcal{A})$ is a *closed ABox repair* of $\mathcal{A}$ w.r.t. $\mathcal{T}$ if (i) it is $\mathcal{T}$-consistent, and (ii) there is no $\mathcal{T}$-consistent $\mathcal{R}' \subseteq \mathsf{close}^*_{\mathcal{T}}(\mathcal{A})$ such that $\mathcal{R} \cap \mathcal{A} \subsetneq \mathcal{R}' \cap \mathcal{A}$ or $\mathcal{R} \cap \mathcal{A} = \mathcal{R}' \cap \mathcal{A}$ and $\mathcal{R} \subsetneq \mathcal{R}'$. If $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, the set of closed ABox repairs of $\mathcal{A}$ w.r.t. $\mathcal{T}$ is denoted $ClosedRep(\mathcal{K})$.

Closed ABox repairs can be seen as maximally 'completing' the (plain) ABox repairs with assertions from $\mathsf{close}^*_{\mathcal{T}}(\mathcal{A}) \setminus \mathcal{A}$. Subsequent studies [46,33] have adopted a different definition of closed ABox repair that simply takes the repairs of the KB $\langle \mathcal{T}, \mathsf{close}^*_{\mathcal{T}}(\mathcal{A})\rangle$. The two definitions do not coincide since a repair of $\langle \mathcal{T}, \mathsf{close}^*_{\mathcal{T}}(\mathcal{A})\rangle$ need not be a closed ABox repair according to the previous definition (see [12] for an example).

Definition 7 (CAR & ICAR semantics) A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ under the *CAR (Closed ABox Repair) semantics* just in the case that $\langle \mathcal{T}, \mathcal{R}\rangle \models q(\mathbf{a})$ for every $\mathcal{R} \in \mathit{ClosedRep}(\mathcal{K})$. It is an answer under the *ICAR (Intersection of Closed ABox Repairs) semantics* if and only if $\langle \mathcal{T}, \mathcal{D}\rangle \models q(\mathbf{a})$ where $\mathcal{D} = \bigcap_{\mathcal{B} \in \mathit{ClosedRep}(\mathcal{K})} \mathcal{B}$.

Remark 1 While the variants of the CAR and ICAR semantics induced by the simpler definition of closed ABox repair may produce different query results, they possess similar computational properties [46,33].

On the KB from Example 1, the CAR (resp. ICAR) semantics gives the same answers as the AR (resp. ICR semantics). We present another example (again borrowed from [12]) to show how these semantics differ.

Example 4 Let $\mathcal{T}'_{\mathsf{univ}}$ be obtained from $\mathcal{T}_{\mathsf{univ}}$ by adding $\exists\mathsf{Teaches} \sqsubseteq \mathsf{Faculty}$. Then $\mathsf{close}^*_{\mathcal{T}'_{\mathsf{univ}}}(\mathcal{A}_{\mathsf{univ}})$ contains $\mathcal{A}_{\mathsf{univ}}$ as well as the following additional assertions:

$\{\mathsf{Faculty}(\mathsf{sam}), \mathsf{Faculty}(\mathsf{kim}), \mathsf{Faculty}(\mathsf{alex}),$
$\mathsf{Faculty}(\mathsf{jane}), \mathsf{Course}(\mathsf{jane}), \mathsf{Faculty}(\mathsf{cs34}), \mathsf{Course}(\mathsf{cs48})\}$

Since $\mathsf{Faculty}(\mathsf{cs34})$ is not involved in any contradictions, it appears in every closed ABox repair, so $\mathsf{cs34}$ is an answer to $q(x) = \mathsf{Faculty}(x)$ under ICAR and CAR semantics. Note however that $\mathsf{cs34}$ is not an answer under AR semantics. This is because some (standard) repairs contain the assertion $\mathsf{Fellow}(\mathsf{jane})$, and hence will not contain the conflicting assertion $\mathsf{Teaches}(\mathsf{cs34}, \mathsf{jane})$, and it is only the latter assertion that allows us to infer $\mathsf{Faculty}(\mathsf{cs34})$.

As displayed in Fig. 1, the CAR semantics satisfies Consistent Results, and ICAR semantics further satisfies Unique Base, but neither semantics satisfies Consistent Support (here again we refer to [12] for a counterexample).

We next consider a parameterized family of semantics, called the k-support semantics, that were introduced in [18] in order to provide increasingly more fine-grained lower approximations of the AR semantics (while enjoying certain desirable computational properties, as discussed in Section 4).

Definition 8 (k-support semantics) Tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ under the *k-support semantics*, written $\langle \mathcal{T}, \mathcal{A}\rangle \models_{k\text{-supp}} q(\mathbf{a})$, if there exist (not necessarily distinct) subsets $S_1, \ldots, S_k$ of $\mathcal{A}$ that satisfy the following:

- each S_i is a $\mathcal{T}$-support for $q(\mathbf{a})$ in $\mathcal{A}$
- for every $R \in \mathit{Rep}(\mathcal{K})$, there is some S_i with $S_i \subseteq R$

The intuition for the k-support semantics is to restrict the number of distinct supports of the query that can be used to 'cover' all of the repairs. When $k = 1$, the same support must be present in every repair, so the 1-support semantics coincides with the IAR semantics. By increasing k and allowing larger and larger supports, the set of answers will increase until it coincides with the AR-answers. Like the AR semantics, the k-support semantics satisfy both Consistent Results and Consistent Support.

Example 5 Continuing our running example, we evaluate $\mathsf{Faculty}(x)$ using the k-support semantics. When $k = 1$, the semantics coincides with the IAR semantics, so we only get alex. For $k = 2$, we gain an additional answer, kim, by considering the pair of supports $\{\mathsf{Prof}(\mathsf{kim})\}$ and $\{\mathsf{Lect}(\mathsf{kim})\}$. Finally, for $k \geq 3$, we have one further answer, sam, by considering the supports $\{\mathsf{Prof}(\mathsf{sam})\}$, $\{\mathsf{Lect}(\mathsf{sam})\}$, $\{\mathsf{Fellow}(\mathsf{sam})\}$.

A second parameterized class of semantics, the k-defeater semantics, was introduced in the same work [18] in order to provide increasingly tighter upper approximations of the AR semantics.

Definition 9 (k-defeater semantics) A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ under the *k-defeater semantics*, written $\mathcal{K} \models_{k\text{-def}} q(\mathbf{a})$, if there does not exist a $\mathcal{T}$-consistent subset S of $\mathcal{A}$ with $|S| \leq k$ such that $\langle \mathcal{T}, S \cup C\rangle \models \bot$ for every inclusion-minimal $\mathcal{T}$-support $C \subseteq \mathcal{A}$ of $q(\mathbf{a})$.

It has been shown in [18] that 0-defeater semantics coincides with brave semantics and that the set of answers under k-defeater semantics decreases as the value of k increases, until the set of AR-answers is reached.

Let us also mention another parameterized family of semantics, called k-lazy [41] semantics, which was originally proposed for Datalog +/- ontologies and whose definition involves another notion of repair (omitted for lack of space). By taking k large enough, the k-lazy semantics coincides with the AR semantics. However, in contrast to the k-support semantics, the convergence is not monotone, i.e. a tuple might be an answer for $k = \ell$ but no longer an answer when $k = \ell + 1$. Due to this behaviour, the k-lazy semantics are

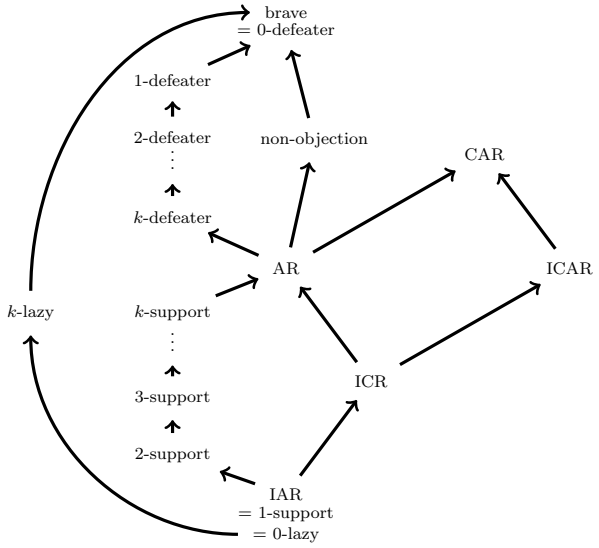

Fig. 2: Relationships between inconsistency-tolerant semantics, where an arrow $S \rightarrow S'$ means that S is an under-approximation of S', i.e., $\langle \mathcal{T}, \mathcal{A} \rangle \models_S q(\mathbf{a}) \Rightarrow \langle \mathcal{T}, \mathcal{A} \rangle \models_{S'} q(\mathbf{a})$.

not always under-approximations of the AR semantics, though they do satisfy the Consistent Results and Consistent Support properties.

Another natural over-approximation of the AR semantics was proposed in [7]:

Definition 10 (Non-objection semantics) A tuple $\mathbf{a}$ is an answer to q over $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ under the *non-objection semantics* if (i) there is some $\mathcal{B} \in \mathit{Rep}(\mathcal{K})$ with $\langle \mathcal{T}, \mathcal{B} \rangle \models q(\mathbf{a})$, and (ii) for every $\mathcal{B} \in \mathit{Rep}(\mathcal{K})$, there is a model of $\langle \mathcal{T}, \mathcal{B} \rangle$ where $q(\mathbf{a})$ is satisfied.

The non-objection semantics lies between the brave and AR semantics, and it satisfies both Consistent Results and Consistent Support.

We summarize the relationships holding between the different semantics discussed thus far in Fig. 2.

As noted at the beginning of the section, repairs are usually defined using set inclusion. However, in some cases, it can be more appropriate to select only the most preferred repairs according to some criteria. Several different notions of preferred repair, based upon cardinality, priority levels, partial preorders, or weighted assertions, have been explored [49,14,28,5,39,6,39] and used as the basis for inconsistency-tolerant semantics.

We further note that the preceding works focused on repairing data given in the form of an ABox (or set of facts over the ontology vocabulary), and the definitions need to be adapted to handle the setting of ontology-based data access (OBDA), where existing data sources are linked to a TBox via mappings. This issue has been explored recently in [11], where two different approaches (repair-at-source, map-then-repair) are contrasted, and it is closely related to consistent query answering in data integration [19,21] and data exchange [23] settings.

Finally, we should emphasize that there is no single 'best' semantics, and the choice of which to use needs to be based upon the acceptable level of risk as well as performance requirements. Moreover, it can be fruitful to utilize multiple semantics in combination, either for computational benefit or to identify answers with different levels of plausibility.

4 Complexity of Querying DL KBs under Inconsistency-Tolerant Semantics

The complexity of query answering under inconsistency-tolerant semantics has been the subject of numerous works. We briefly present what is known for DL ontologies and refer to [12] for further details and references to where the stated results were proven.

We recall that there are two standard ways of measuring the complexity of query answering. *Combined complexity* is w.r.t. the size of the whole input (TBox, ABox, query), while *data complexity* is measured w.r.t. the size of ABox (with the query and TBox treated as fixed). Our results make reference to the well-known complexity classes P, NP, and coNP, as well as the following classes whose definitions we recall: AC^0 (problems that can be solved by a uniform family of circuits of constant depth and polynomial size, with unlimited fan-in AND gates and OR gates), NL (problems solvable in non-deterministic logarithmic space), $\Delta_2^p[O(\log n)]$(problems solvable in polynomial time with at most logarithmically many calls to an NP oracle), and Π_2^p(problems whose complement is solvable in non-deterministic polynomial time with access to an NP oracle). The classes AC^0 and NL are contained in P.

Let us start by the most well-studied case, namely, DL-Lite KBs. Fig. 3 displays the complexity landscape for querying DL-Lite[1] KBs under various inconsistency-tolerant semantics, considering both data and combined complexity measures and both conjunctive queries and instance queries. We observe that there are several semantics for which query answering is in AC^0 in data complexity. These upper bounds are shown by means of first-order query rewriting. For the IAR semantics, the rough idea is to modify a usual rewriting by adding negated atoms that forbid the use of ABox assertions that do not belong to the intersection of repairs [33,9]. Subsequent work [18] established general rewritability

[1] The results apply to common DL-Lite dialects, such as DL-Lite$_{\mathsf{core}}$, DL-Lite$_{\mathcal{R}}$, and DL-Lite$_{\mathcal{A}}$, see [12] for details.

Semantics	CQs		IQs	
	data	comb	data	comb
classical	in AC^0	NP	in AC^0	NL
AR	coNP	Π_2^p	coNP	coNP
IAR	in AC^0	NP	in AC^0	NL
brave	in AC^0	NP	in AC^0	NL
ICR	coNP	$\Delta_2^p[O(log\, n)]$	coNP	coNP
CAR	coNP	Π_2^p	in AC^0	NL
ICAR	in AC^0	NP	in AC^0	NL
k**-support**	in AC^0	NP	in AC^0	NL
k**-defeater**	in AC^0	NP	in AC^0	NL
k**-lazy**	coNP	Π_2^p	in P	in P
non-objection	in AC^0	NP	in AC^0	NL

Fig. 3: Data complexity (data) and combined complexity (comb) of CQ and IQ answering over DL-Lite KBs. All results are completeness results unless otherwise indicated.

DL	Semantics	Data	Combined
$\mathcal{EL}_\bot$	**classical**	P	NP
	AR	coNP	Π_2^p
	IAR	coNP	$\Delta_2^p[O(log\, n)]$
	brave	NP	NP
$\mathcal{ALC}$	**classical**	coNP	Exp
	AR	Π_2^p	Exp
	IAR	Π_2^p	Exp
	brave	Σ_2^p	Exp

Fig. 4: Complexity of answering CQs in $\mathcal{EL}_\bot$ and $\mathcal{ALC}$. All results are completeness results.

results that apply to the families of k-support and k-defeater semantics and arbitrary FO-rewritable ontology languages. We note that the AC^0 result for non-objection semantics[2] has not been stated in the literature but can be shown by adapting query rewriting techniques for the brave and IAR semantics. For the AR semantics, which is arguably the most natural, query answering is intractable in data complexity, even in restricted settings, like IQs [32] or very simple TBoxes (a single disjointness axiom $T \sqsubseteq \neg F$ suffices [10]). Turning now to combined complexity, we observe that the semantics that are well behaved for data complexity remain so for combined complexity (i.e. their complexity matches that of classical semantics), while the semantics with intractable data complexity exhibit higher combined complexities than classical semantics. Finally, we note that the complexity of querying with variants of AR and IAR based upon preferred repairs (cardinality, weights, priorities) has also been studied (see e.g. [14]), and the general message is that incorporating preferences leads to higher complexity.

We now briefly consider the situation for DLs beyond DL-Lite. Fig. 4 displays complexity results for two representative DLs (the lightweight DL $\mathcal{EL}_\bot$ and the expressive DL $\mathcal{ALC}$) and three prominent semantics (AR, IAR, brave). The results for the AR and IAR semantics were established in [46], while those for brave semantics can be found in [12]. The main observation with regards to $\mathcal{EL}_\bot$ (and other Horn DLs) is that the IAR and brave semantics are no longer tractable in data complexity. Essentially, the reason is that in contrast to DL-Lite and other FO-rewritable languages, it is not possible in general to bound the size of minimal $\mathcal{T}$-supports nor minimal $\mathcal{T}$-inconsistent subsets (i.e., subsets of the ABox that are $\mathcal{T}$-inconsistent but whose every proper subset is $\mathcal{T}$-consistent). For the expressive DL $\mathcal{ALC}$, the adoption of inconsistency-tolerant semantics leads to a rise in data complexity, but leaves the combined complexity unchanged (since the repairs can be enumerated in exponential time).

5 Implementing Inconsistency-Tolerant OMQA

We give a brief overview of systems that have been implemented and tested for inconsistency-tolerant query answering over DL knowledge bases.

QuID system [47, 34] This system[3] performs conjunctive query answering under the IAR semantics in an extension of DL-Lite$_\mathcal{A}$ with denial and identification constraints. Three different approaches have been implemented and compared: first-order query rewriting, ABox annotation (in which assertions are marked as safe or problematic depending on whether they belong to the intersection of repairs, and the query is modified to only use safe assertions), and ABox cleaning (in which assertions not belonging to the intersection of repairs are removed, and the resulting dataset is queried as usual). The latter two approaches generally proved to be more efficient than the rewriting approach, but they have the downside of involving data modifications.

CQAPri system [14, 17] This system[4] computes answers to CQs over DL-Lite$_\mathcal{R}$ KBs under the IAR, brave, and AR semantics (as well as prioritized versions of AR and IAR). Answers are first computed for the IAR and brave semantics, by evaluating a UCQ-rewriting and filtering the results using a pre-computed set of minimal $\mathcal{T}$-inconsistent subsets. To identify the AR-answers

[2] In [7], only polynomial data complexity is proven, which we improve to AC^0. It is also not too hard to show that the combined complexity matches classical semantics.

[3] QuID: `www.dis.uniroma1.it/~ruzzi/quid/`

[4] CQAPri: `www.lri.fr/~bourgaux/CQAPri`.

among the remaining tuples (i.e. those holding under brave semantics but not under IAR semantics), CQAPri constructs a (usually quite small) instance of UNSAT for every such tuple, which is passed to an off-the-shelf SAT solver. In addition to using the IAR and brave semantics to reduce the number of calls to the SAT solver, the three semantics are used to partition query answers into three levels of reliability: (Almost) Sure (those answers holding under IAR semantics), Likely (answers holding under AR but not IAR semantics), and Possible (answers only holding under brave semantics). Experiments conducted on the modified LUBM benchmark (which was further augmented with negative inclusions and conflicting assertions) showed that despite its intractable data complexity, it is feasible to compute query answers under the AR semantics, thanks in part to the fact that many AR-answers can be identifed using the tractable IAR semantics.

SaQAI system [51] This system[5] implements the IAR and ICAR semantics for DL-Lite$_\mathcal{R}$ KBs and CQs. For the IAR semantics, the authors follow the ABox cleaning approach from QuID, using query rewriting to identify and then remove the assertions that do not appear in the intersection of repairs. For the ICAR semantics, a combination of saturation and query rewriting is employed, together with some optimizations. The experiments conducted using the CQAPri benchmark show a better performance than the QuID and CQAPri systems for the IAR semantics.

System from [50] This system targets the IAR semantics and currently supports the DL $\mathcal{ELH}^{dr}_\bot$. It checks whether the sufficient conditions for producing a rewriting w.r.t. IAR semantics are fulfilled (by making calls to the FO-rewritability checking system Grind [29]) and constructs such a rewriting when one exists by adding negated conjuncts to a classical rewriting. Experiments were conducted on seven existing ontologies (which sometimes needed to be enriched with negative inclusions to allow for inconsistencies) and for six of them, the sufficient conditions were satisfied, suggesting that a rewriting-based approach to IAR may be feasible in practice for ontologies beyond DL-Lite.

System from [7] This system implements the non-objection semantics for ground CQs (i.e. CQs without existentially quantified variables) for DL-Lite$_\mathcal{R}$ KBs. Experiments on the CQAPri benchmark confirm that query answers can be efficiently computed (in accordance with the tractable data complexity).

[5] SaQAI: www.image.ece.ntua.gr/~etsalap/SaQAI/

System from [28] This system can be utilized to query $\mathcal{SHIQ}$ KBs under a variant of the AR semantics in which ABox assertions are assigned weights and querying is restricted to ground CQs. Like CQAPri, it employs SAT solvers as well as a form of reachability analysis to identify a query-relevant fragment of the KB.

6 Related Reasoning Services for Inconsistency Handling

We mention some related reasoning and analysis tasks. First, to render inconsistency-tolerant OMQA systems more usable, it is important to be able to explain the results to users. This issue has been taken up in [15], where a formal framework was presented for justifying why a given tuple appears as an answer under the considered inconsistency-tolerant semantics (AR, IAR, or brave) or why it is not part of the results. The approach has been implemented by exploiting different functionalities of SAT solvers and integrated into the CQAPri system. Closely related is a line of work [3,2] on utilizing argumentation and dialogues with users to explain query answers under various inconsistency-tolerant semantics (ICR, IAR, brave, and AR).

Another important question is how to aid users in repairing their data, in order to improve the quality of the data. An interactive query-driven approach to this question has been presented in [16]. The idea is to allow users to provide feedback on which query results are missing or erroneous, and then interact with the user in order to identify a set of ABox modifications (additions and deletions of assertions) that fix the identified flaws. The ABox update problem [27,38] is also concerned with modifying the ABox to ensure consistency, but does not involve interaction with a user and targets a setting in which inconsistencies result from changes to the actual state of affairs.

While we have assumed in this paper that errors originate from the data, this presupposes that the ontology has been properly debugged. Several different axiom pinpointing and justification finding algorithms [48,30,45] have been proposed to aid ontology engineers in identifying the sources of unwanted inferences. An approach to repairing DL KBs to be able to infer missing consequences while avoiding some undesired entailments has been presented in [43]. In the OBDA setting, recent works have examined reasoning tasks such as checking whether the mapping is coherent w.r.t. the ontology [35], minimally modifying a mapping to reflect changes to the database schema or ontology [36], and deciding if a database schema protects an OBDA specification [25], i.e., every legal data instance for the

database constraints is consistent w.r.t. the ontology and mapping.

Finally, let us also point out that some close connections have been identified between inconsistency-tolerant OMQA and the areas of argumentation [26, 13] and privacy-aware query evaluation [37].

7 Concluding Remarks and Future Work

We hope to have showcased the large body of research that has been developed over the past decade or so around the issue of inconsistency handling in OMQA. Significant progress has been made on proposing different semantics for querying inconsistent KBs in a principled manner and exploring their computational properties: complexity, algorithms, and implemented prototypes. There nevertheless remain several interesting theoretical and practical challenges to tackle going forward, let us mention just three.

First, while we start to have a reasonable idea of how to approach the problem for DL-Lite KBs, there remains a need to develop practical algorithms for DLs beyond DL-Lite. Indeed, due to the prevalence of data quality issues, *every OMQA system should be equipped with some sort of inconsistency handling mechanism* (beyond simply reporting that the KB is inconsistent!), and the challenge is to find ways of incorporating such features while limiting the impact on performance. First steps towards this goal can be found in [51,50].

Second, a very nice but extremely challenging theoretical question is to classify the complexity of inconsistency-tolerant query answering at the level of ontology-mediated queries (that is, ontology-query pairs). Some preliminary results in this direction have been presented in [9,10]. We note that this problem is closely related to work on classifying the complexity of consistent query answering in the presence of functional dependencies, where significant progress has been made (see e.g. [31]), but a full classification has proven elusive.

Third, it would also be worthwhile to develop quantitative approaches to inconsistency-tolerant OMQA, both to be able to quantify the confidence in different results, and to be able to take advantage of numeric / probabilistic / statistical information when it is available. For instance, data that results from information extraction systems is often annotated with a confidence value, and mined data quality rules (see e.g. [44]) that act as soft constraints can prove useful in detecting inconsistencies and determining the most likely fixes.

Acknowledgements The author would like to thank Camille Bourgaux, who was a co-author of the survey chapter [12] and contributed to the running example reproduced here.

References

1. Arenas, M., Bertossi, L.E., Chomicki, J.: In: Proceedings of PODS, pp. 68–79
2. Arioua, A., Croitoru, M.: Dialectical characterization of consistent query explanation with existential rules. In: Proceedings of FLAIRS (2016)
3. Arioua, A., Tamani, N., Croitoru, M.: Query answering explanation in inconsistent Datalog +/- knowledge bases. In: Proceedings of DEXA (2015)
4. Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic. Cambridge University Press (2017)
5. Baget, J., Benferhat, S., Bouraoui, Z., Croitoru, M., Mugnier, M., Papini, O., Rocher, S., Tabia, K.: Inconsistency-tolerant query answering: Rationality properties and computational complexity analysis. In: Proceedings of JELIA, pp. 64–80 (2016)
6. Belabbes, S., Benferhat, S.: Inconsistency handling for partially preordered ontologies: Going beyond Elect. In: Proceedings of KSEM (2019)
7. Benferhat, S., Bouraoui, Z., Croitoru, M., Papini, O., Tabia, K.: Non-objection inference for inconsistency-tolerant query answering. In: Proceedings of IJCAI, pp. 3684–3690 (2016)
8. Bertossi, L.E.: Database Repairing and Consistent Query Answering. Synthesis Lectures on Data Management. Morgan & Claypool Publishers (2011)
9. Bienvenu, M.: First-order expressibility results for queries over inconsistent DL-Lite knowledge bases. In: Proceedings of DL Workshop (2011)
10. Bienvenu, M.: On the complexity of consistent query answering in the presence of simple ontologies. In: Proceedings of AAAI (2012)
11. Bienvenu, M.: Inconsistency-tolerant ontology-based data access revisited: Taking mappings into account. In: Proceedings of IJCAI, pp. 1721–1729 (2018)
12. Bienvenu, M., Bourgaux, C.: Inconsistency-tolerant querying of description logic knowledge bases. In: Reasoning Web Tutorial Lectures, pp. 156–202 (2016)
13. Bienvenu, M., Bourgaux, C.: Querying and repairing inconsistent prioritized knowledge bases: Complexity analysis and links with abstract argumentation. In: Proceedings of KR (2020)
14. Bienvenu, M., Bourgaux, C., Goasdoué, F.: Querying inconsistent description logic knowledge bases under preferred repair semantics. In: Proceedings of AAAI (2014)
15. Bienvenu, M., Bourgaux, C., Goasdoué, F.: Explaining inconsistency-tolerant query answering over description logic knowledge bases. In: Proceedings of AAAI (2016)
16. Bienvenu, M., Bourgaux, C., Goasdoué, F.: Query-driven repairing of inconsistent DL-Lite knowledge bases. In: Proceedings of IJCAI (2016)
17. Bienvenu, M., Bourgaux, C., Goasdoué, F.: Computing and explaining query answers over inconsistent DL-Lite knowledge bases. Journal of Artificial Intelligence Research **64**, 563–644 (2019)
18. Bienvenu, M., Rosati, R.: Tractable approximations of consistent query answering for robust ontology-based data access. In: Proceedings of IJCAI (2013)
19. Bravo, L., Bertossi, L.E.: Logic programs for consistently querying data integration systems. In: Proceedings of IJCAI, pp. 10–15 (2003)
20. Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. Journal of Web Semantics **14**, 57–83 (2012)

21. Calì, A., Lembo, D., Rosati, R.: Query rewriting and answering under constraints in data integration systems. In: Proceedings of IJCAI, pp. 16–21 (2003)
22. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. Journal of Automated Reasoning **39**(3), 385–429 (2007)
23. ten Cate, B., Halpert, R.L., Kolaitis, P.G.: Exchange-repairs - Managing inconsistency in data exchange. Journal of Data Semantics **5**(2), 77–97 (2016)
24. Chomicki, J.: Consistent query answering: Five easy pieces. In: Proceedings of ICDT, pp. 1–17 (2007)
25. Console, M., Lenzerini, M.: Data quality in ontology-based data access: The case of consistency. In: Proceedings of AAAI, pp. 1020–1026 (2014)
26. Croitoru, M., Vesic, S.: What can argumentation do for inconsistent ontology query answering? In: Proceedings of SUM (2013)
27. De Giacomo, G., Lenzerini, M., Poggi, A., Rosati, R.: On instance-level update and erasure in description logic ontologies. Journal of Logic and Computation **19**(5), 745–770 (2009)
28. Du, J., Qi, G., Shen, Y.D.: Weight-based consistent query answering over inconsistent $\mathcal{SHIQ}$ knowledge bases. Knowledge and Information Systems **34**(2), 335–371 (2013)
29. Hansen, P., Lutz, C., Seylan, I., Wolter, F.: Efficient query rewriting in the description logic $\mathcal{EL}$ and beyond. In: Proceedings of IJCAI, pp. 3034–3040 (2015)
30. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A.: Debugging unsatisfiable classes in OWL ontologies. Journal Web Sem. **3**(4), 268–293 (2005)
31. Koutris, P., Wijsen, J.: Consistent query answering for self-join-free conjunctive queries under primary key constraints. ACM Transactions on Database Systems **42**(2), 9:1–9:45 (2017)
32. Lembo, D., Lenzerini, M., Rosati, R., Ruzzi, M., Savo, D.F.: Inconsistency-tolerant semantics for description logics. In: Proceedings of RR (2010)
33. Lembo, D., Lenzerini, M., Rosati, R., Ruzzi, M., Savo, D.F.: Query rewriting for inconsistent DL-Lite ontologies. In: Proceedings of RR (2011)
34. Lembo, D., Lenzerini, M., Rosati, R., Ruzzi, M., Savo, D.F.: Inconsistency-tolerant query answering in ontology-based data access. Journal of Web Semantics **33**, 3–29 (2015)
35. Lembo, D., Mora, J., Rosati, R., Savo, D.F., Thorstensen, E.: Mapping analysis in ontology-based data access: Algorithms and complexity. In: Proceedings of ISWC, pp. 217–234 (2015)
36. Lembo, D., Rosati, R., Santarelli, V., Savo, D.F., Thorstensen, E.: Mapping repair in ontology-based data access evolving systems. In: Proceedings of IJCAI, pp. 1160–1166 (2017)
37. Lembo, D., Rosati, R., Savo, D.F.: Revisiting controlled query evaluation in description logics. In: Proceedings of IJCAI, pp. 1786–1792 (2019)
38. Liu, H., Lutz, C., Milicic, M., Wolter, F.: Foundations of instance level updates in expressive description logics. Artificial Intelligence **175**(18), 2170–2197 (2011)
39. Lukasiewicz, T., Malizia, E., Vaicenavicius, A.: Complexity of inconsistency-tolerant query answering in Datalog+/- under cardinality-based repairs. In: Proceedings of AAAI (2019)
40. Lukasiewicz, T., Martinez, M.V., Pieris, A., Simari, G.I.: From classical to consistent query answering under existential rules. In: Proceedings of AAAI, pp. 1546–1552 (2015)
41. Lukasiewicz, T., Martinez, M.V., Simari, G.I.: Inconsistency handling in Datalog+/- ontologies. In: Proceedings of ECAI (2012)
42. Mugnier, M., Thomazo, M.: An introduction to ontology-based query answering with existential rules. In: Reasoning Web Tutorial Lectures, pp. 245–278 (2014)
43. Nikitina, N., Rudolph, S., Glimm, B.: Interactive ontology revision. Journal of Web Semantics **12**, 118–130 (2012)
44. Ortona, S., Meduri, V.V., Papotti, P.: Rudik: Rule discovery in knowledge bases. Proceedings of PVLDB **11**(12), 1946–1949 (2018)
45. Peñaloza, R., Sertkaya, B.: Complexity of axiom pinpointing in the DL-Lite family of description logics. In: Proceedings of ECAI (2010)
46. Rosati, R.: On the complexity of dealing with inconsistency in description logic ontologies. In: Proceedings of IJCAI (2011)
47. Rosati, R., Ruzzi, M., Graziosi, M., Masotti, G.: Evaluation of techniques for inconsistency handling in OWL 2 QL ontologies. In: Proceedings of ISWC (2012)
48. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proceedings of IJCAI, pp. 355–362 (2003)
49. Staworko, S., Chomicki, J., Marcinkowski, J.: Prioritized repairing and consistent query answering in relational databases. Annals of Mathematics and Artificial Intelligence **64**(2-3), 209–246 (2012)
50. Trivela, D., Stoilos, G., Vassalos, V.: A framework and positive results for IAR-answering. In: Proceedings of AAAI (2018)
51. Tsalapati, E., Stoilos, G., Stamou, G.B., Koletsos, G.: Efficient query answering over expressive inconsistent description logics. In: Proceedings of IJCAI (2016)