On the Capabilities and Limitations of OWL Regarding Typecasting and Ontology Design Pattern Views

Adila A. Krisnadhi[1,3], Pascal Hitzler[1], and Krzysztof Janowicz[2]

[1] Wright State University, Dayton, OH, USA
[2] University of California, Santa Barbara, CA, USA
[3] Universitas Indonesia, Depok, Indonesia

Abstract. In ontology engineering, particularly when dealing with heterogeneous domains and their subfields, legacy data, various data models, existing standards, code lists, and so forth, there is a frequently recurring need to express certain types of axioms that allow diverse representational choices interoperate. Some of these axioms, which we call typecasting axioms, point to limitations of the Web Ontology Language (OWL), while others require best practice guides for the community. Here, we introduce these typecasting axioms and elaborate how such axioms can help the development of data integration using ontology and ontology patterns. We then conclude with a brief catalog of open research problems motivated by typecasting axioms, which may be of potential interest to both application developers and researchers working on logical foundations of OWL.

1 Introduction and Motivation

During our ontology engineering work with subject matter experts from a wide range of domains including the broader geo-sciences [10], industrial ecology, the digital humanities, libraries and the publishing industry, particle physics, and so forth, we became aware of the recurring need to express certain types of axioms necessary to bridge diverse representational choices, and thus enable interoperability between them in the same ontological framework. Often, these axioms can be easily expressed using first-order predicate logic, but the description logic (DL) underlying the Web Ontology Language OWL [7] does not – or not obviously – enable us to express these axioms.

Our goal is twofold: first, to motivate and describe these types of axioms as well as the capabilities and limitations of OWL to represent them, to the extent we are aware of them; and second, to highlight these limitations as open problems on which researchers interested in improving and extending OWL and its underlying logics could work. In this paper, we particularly focus on axioms called *typecasting* axioms, which allow one to seamlessly switch between class-centric, individual-centric, and property-centric representation. These axioms are not only relevant to ontology modeling, but also in ontology alignment with complex mappings. Some of the issues herein were already alluded to by Noy [16], however they were not discussed in the context of formal semantics of OWL and the underlying description logics-based formalisms.

Our discussion is structured as follows. Section 2 introduces typecasting axioms and discusses how and why OWL can or cannot represent them. This is followed by Section 3 where we further discuss a particular use of typecasting axioms in the context of Ontology Design Patterns [6] to express *view* expansion and contraction, which are useful in the context of linked data publishing and integration [10,12,17]. We then conclude in Section 4 with a list of research questions compiled from throughout the paper.

2 Typecasting in OWL

In this section we discuss three kinds of typecasting that we frequently encountered in our work on ontological modeling. Casting between types is the implicit or explicit process by with one (data) type is converted into another type, e.g., widening an *int* to a *long*. In object-oriented modeling, this includes accessing objects that instantiate certain types as objects of their common ancestor type. To give a simple example, a Point Of Interest (POI) class may define a method to return the spatial footprint of the place as geographic (point) coordinates. All classes that extend the POI class, say Restaurant and Hospital, can be queried for their footprint by iterating over a collection of POI.

Here, we use the term typecasting to refer to translation between multiple representational choices to define a notion using either individuals, classes, or properties in the context of description logics. This is part of a bigger picture that we call *ontology virtualization* by which an underlying model can be exposed in different ways to suit particular needs or paradigms.

2.1 Typecasting Individual to Class and Back: Explicit Versus Implicit Typing of Instances

The first case is concerned with the representational choice between the explicit typing of individuals (via rdf:type) versus the identification of the type of an individual by reference to a classname, given as an individual. In other words, we do typecasting from a class to an individual and vice versa.

Problem Description Schematically, the two representational choices are depicted in Figure 1. Case 1a at the top part of the figure corresponds to the explicit use of `rdf:type` to assert a type of an individual, which, generally speaking, seems to be more in the spirit of OWL. Case 1b at the bottom part of the figure, however, sometimes seem more natural for domain experts, e.g., when referring to an externally controlled vocabulary.

Consider, for example, the case of measurement types. A concrete measurement of a particular characteristic of a feature of interest, e.g., a lake, can be

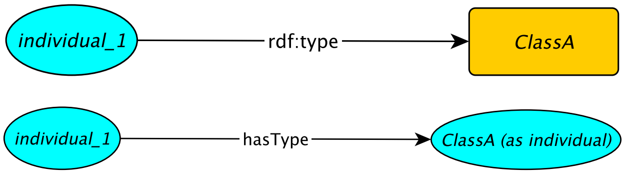

Fig. 1. Axiom (1) maps Case 1a (top part) to Case 1b (bottom part), while axiom (2) maps in the opposite direction. Blue nodes are OWL named individuals, while the yellow node is an OWL class.

of type NitrateConcentration (which in turn is a Concentration measurement). This can be asserted through the following triple.

```
ex:measurement1  rdf:type  geo:NitrateConcentration .
```

We assume here that the namespace `geo:` refers to an appropriate ontology that contains measurement types. At the same time, however, it may be appropriate for this ontology to incorporate an existing controlled vocabulary for the identification of measurement types widely used in certain fields of the geosciences. Such controlled vocabularies often come in the form of code lists or may describe measurement types as individuals because one may want to give additional information about them, say

```
geo:NO3-concentration  geo:potentialSource geo:urban_runoff .
```

With `geo:NO3-concentration` being an individual, one would therefore appropriately identify the measurement type for `:measurement1` by specifying

```
ex:measurement1 geo:hasMeasurementType geo:NO3-concentration .
```

Logical Aspects This typecasting case can be handled easily in OWL. Axiom (1) maps Case 1a to 1b, while axiom (2) maps Case 1b to 1a.

$$\text{ClassName} \sqsubseteq \exists \text{hasType}.\{\text{classname}\} \tag{1}$$

$$\exists \text{hasType}.\{\text{classname}\} \sqsubseteq \text{ClassName} \tag{2}$$

For the nitrate concentration example, the mappings in both direction can be expressed by:

$$\texttt{geo:NitrateConcentration} \equiv \exists\texttt{geo:hasMeasurementType}.\{\texttt{geo:NO3-concentration}\}.$$

It shall be noted that the above case is closely related to punning between classes and individuals, i.e., the use of one identifier to denote both a class name and an individual name, which is allowed in OWL 2 DL. In fact, in the above example, we could have used `geo:NO3-concentration` as a class name in addition to using it as an individual name (or only using `geo:NitrateConcentration` for

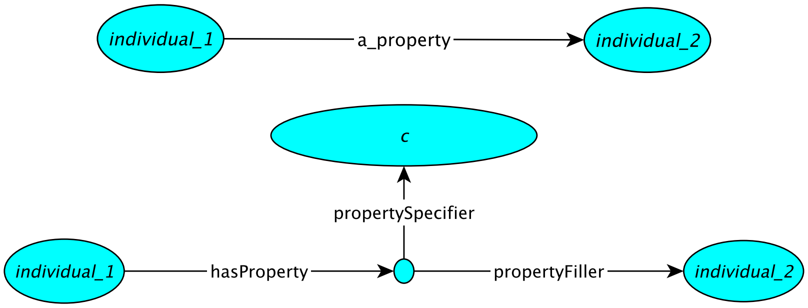

Fig. 2. Case 2a (top) and Case 2b (bottom). In the bottom part, the unlabeled blue node can either be a named or anonymous individual (blank node).

that matter). Typecasting is, however, more general in the sense that it allows one to use different identifier to refer to the same typing of an instance and employ any object property as a typing predicate, simulating `rdf:type`.

2.2 Typecasting between Class and Property

The next two kinds of typecasting is concerned with the representational choice between using a simple property or using a class to represent a relationship between two entities.

Problem Description. Schematically, the two representational choices are depicted in Figure 2. Here, Case 2a simply uses a property to represent a relationship between two individuals, while Case 2b uses a reified representation of the relation, which is actually a (possibly non-atomic) class.

For example, consider the set of triples in Figure 3 stating that an oceanographic cruise has a lead scientist provided as an individual `ex:PeterWiebe`. These triples in Turtle syntax corresponds to Case 2a (disregarding the instance typing triples).

```
ex:cruise123   geo:hasLeadScientist   ex:PeterWiebe ;
               rdf:type               geo:Cruise .
ex:PeterWiebe  rdf:type               geo:Person .
```

Fig. 3. Triples for Case 2a

Meanwhile, the set of triples in Figure 4 corresponds to Case 2b where essentially the same relationship as above is represented using a reified representation.

```
ex:cruise123  rdf:type   geo:Cruise .
ex:cruise123  geo:hasParticipation  [
                         rdf:type geo:LeadScientistRole ;
                         geo:playsRole  geo:LeadScientist ;
                         geo:isPlayedBy  ex:PeterWiebe ] .
ex:PeterWiebe rdf:type   geo:Person .
```

Fig. 4. Triples for Case 2b

The representational choice as described by Case 2b is in fact very common, e.g. as part of the so-called Agent Role ontology design pattern,[1] and has in some form even be adopted by schema.org under the term *role*.[2] The advantage of the second, more verbose representation is that additional information can be added to the blank node, e.g., the funding agency, affiliation, and so forth. Note that the typecasting discussed in Section 2.1 can also be applied to the individual acting as the property specifier in case 2b above.

Logical Aspects: Rolification Mapping from Case 2b to 2a corresponds to typecasting from class to property. Note that Case 2b is indeed a class-centric representation because the intended relationship is represented through the unlabeled node in the middle of the bottom part of Figure 2, which is an instance of the non-atomic class expression $\exists$propertySpecifier.$\{c\}$. Obviously, one can assign a class name to such an expression if so desired. Also, `rdf:type` and a class name can be used instead of propertySpecifier and c.

The typecasting from class to property desired above employs a technique called *rolification* [11,18].[3] This is a key technique for representation of (Datalog) rules in OWL. There are of course rules that can be readily expressed using OWL axioms, e.g., guarded domain restrictions such as

$$\mathrm{Person}(y) \wedge \mathrm{hasLeadScientist}(x, y) \rightarrow \mathrm{Cruise}(x),$$

which is equivalent to

$$\exists \mathrm{hasLeadScientist}.\mathrm{Person} \sqsubseteq \mathrm{Cruise}$$

However, this is not the case for other rules such as:

$$\begin{aligned}\mathrm{Cruise}(x) \wedge \mathrm{hasParticipation}(x, y) \wedge \mathrm{LeadScientistRole}(y) \wedge \mathrm{isPlayedBy}(y, z)\\ \wedge\, \mathrm{Person}(z) \rightarrow \mathrm{hasLeadScientist}(x, z)\end{aligned}$$

The above rule is in fact what we need to map Case 2b to 2a. It can be expressed in OWL by *rolifying* the three class names, i.e. by introducing new properties R_{Cruise}, $R_{\mathrm{LeadScientistRole}}$, and R_{Person} and asserting the following axioms:

[1] `http://ontologydesignpatterns.org/wiki/Submissions:AgentRole`

[2] `https://schema.org/Role`

[3] In DL literature, properties are called roles, hence the term rolification; not to be confused with role as defined by the Agent Role pattern or schema.org.

$$\mathrm{Cruise} \equiv \exists R_{\mathrm{Cruise}}.\mathrm{Self}, \qquad \mathrm{Person} \equiv \exists R_{\mathrm{Person}}.\mathrm{Self}$$
$$\mathrm{LeadScientistRole} \equiv \exists R_{\mathrm{LeadScientistRole}}.\mathrm{Self}$$
$$R_{\mathrm{Cruise}} \circ \mathrm{hasParticipation} \circ R_{\mathrm{LeadScientistRole}} \circ \mathrm{isPlayedBy} \circ R_{\mathrm{Person}} \sqsubseteq \mathrm{hasLeadScientist}$$

Notice that the three class names were typecasted into property names through rolification axioms of the form $A \equiv \exists R_A.\mathrm{Self}$.

More generally, mapping Case 2b to 2a in Figure 2 is expressed using the following rule

$$\mathrm{hasProperty}(x, y) \wedge \mathrm{propertySpecifier}(y, c) \wedge \mathrm{propertyFiller}(y, z) \\ \rightarrow \mathrm{a\_property}(x, z),$$

which can be expressed in OWL using the two axioms

$$\exists \mathrm{propertySpecifier}.\{c\} \sqsubseteq \exists \mathrm{propertySpecifier\_c}.\mathrm{Self}$$
$$\mathrm{hasProperty} \circ \mathrm{propertySpecifier\_a} \circ \mathrm{propertyFiller} \sqsubseteq \mathrm{a\_property}$$

where the complex class $\exists$propertySpecifier.$\{c\}$ is typecasted into the property propertySpecifier_c via a rolification axiom.

Since rolification allows one to typecast class into property, an additional benefit of the use of rolification is that it allows us to express *typed property chains* of the form

$$R_1 \circ \ldots \circ R_n \sqsubseteq R,$$

where each R_i is either a property name or a class name, the latter of which is to be typecasted into a property using rolification.

There is however a caveat in using rolification axiom. While it is of course expressible in OWL DL, its primary use cases, i.e., general conversion of rules, always involve property chains. For them, OWL DL imposes a so-called *regularity restriction* on the use of property chains [8], which may be violated by the introduced ones. The origin of the regularity restriction is that without it, reasoning over the logic would be undecidable. While this means that the restriction cannot be lifted in its entirety without rendering the logic undesirable, it would be helpful to soften it, i.e., to describe types of cases which violate regularity, but which retain decidability. In addition, approximate work-arounds are possible, e.g., using so-called *nominal schemas* [11,13].

Logical Aspects: Reification Mapping in the other direction, i.e., from case 2a to case 2b, corresponds to typecasting property to class. For example, given a set of triples in Figure 3, we want to express an axiom that allows us to infer triples in Figure 4. This amounts to a well-known modeling technique called *reification.* The axiom cannot be expressed in OWL, but can be handled using rules with existential head – well-known in database as tuple-generating dependencies (TGDs) [2]. In the context of Figure 2, the TGD is of the following form:

$$\mathrm{a\_property}(x, z) \rightarrow \exists y.(\mathrm{hasProperty}(x, y) \wedge \mathrm{propertySpecifier}(y, a) \\ \wedge \mathrm{propertyFiller}(y, z))$$

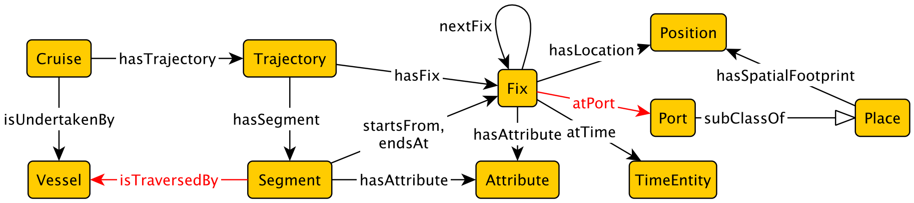

Fig. 5. Part of the GeoLink oceanography ontology to illustrate views.

Note that using rolification axioms is not sufficient because they cannot allow us to infer the existence of the new node for the reification (the RDF blank node in Figure 4).

3 Ontology Design Pattern View Contraction and Expansion

When developing ontologies or ontology design patterns (ODPs) for the purpose of data integration, ontology engineers often have to introduce complex structures like reified relationships to cover the richness of the data being integrated or to provide flexibility in the integrating schema. However, from the perspective of a particular user or data provider, such complications may not be desirable. For them, simplified version of the global schema, which can be specially tailored to be sufficient for their needs may be preferable. In the context of ontology-based or ODP-based data integration, such a simplified version of the global schema corresponds to what we call a *view* by which we mean a set of shortcuts through an ontology or an ontology design pattern. To illustrate the concept, which is discussed also in [12,17], we adapt an example from the GeoLink oceanography ontology [10].

Referring to Figure 5, the red arrows indicate shortcuts, and we will discuss the case of the isTraversedBy shortcut. Of course the picture is only a visualization of a part of the ontology, which consists of a set of OWL axioms which we do not list here.

In the fabric of the ontology, the isTraversedBy shortcut is in fact redundant, i.e. it can be inferred using the rule

$$\begin{aligned}\text{Vessel}(x) \wedge \text{isUndertakenBy}(y,x) \wedge \text{Cruise}(y)\\ \wedge \text{hasTrajectory}(y,z) \wedge \text{Trajectory}(z) \wedge \text{hasSegment}(z,w)\\ \wedge \text{Segment}(w) \rightarrow \text{isTraversedBy}(w,x).\end{aligned} \quad (3)$$

Since the application of the rule results in a simpler representation of the relationship between a trajectory segment and the vessel traversing it, we refer to this type of rule also as a *contraction*.

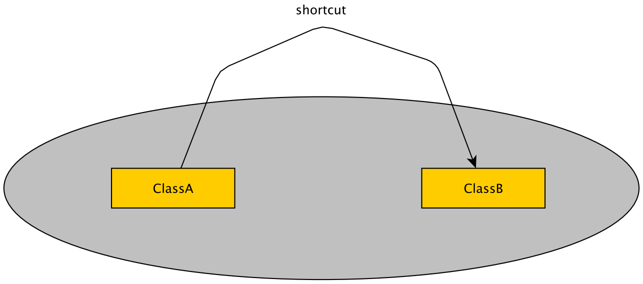

Fig. 6. Depiction of a generic shortcut.

The reverse of a contraction is an *expansion*. In our experience, this case occurs when, e.g., a data provider may have only information about trajectories (and their segments) which oceanographic vessels have taken. In order to populate the ontology with this data, it is required to *expand* the data by inserting an additional individual (or a blank node) as the cruise connecting the trajectory and the vessel.

3.1 Contraction

A generic depiction of the view idea is presented in Figure 6. The grey ellipse shall indicate a labeled graph which in turn can be represented as a conjunction of unary and binary predicates involving ClassA and ClassB, like

$$\mathrm{ClassA}(x) \wedge \mathrm{ClassB}(y) \wedge C_1(x_1) \wedge \cdots \wedge C_n(x_n) \wedge R_1(y_1, y_2) \wedge \cdots \wedge R_k(y_k, y_{k+1}),$$

where x, y, the x_i and the y_j are any variables.

Contraction (i.e., a shortcut between the classes ClassA and ClassB) can then be expressed using the rule

$$\mathrm{ClassA}(x) \wedge \mathrm{ClassB}(y) \wedge C_1(x_1) \wedge \cdots \wedge C_n(x_n) \wedge R_1(y_1, y_2) \wedge \cdots \wedge R_k(y_k, y_{k+1}) \\ \to \mathrm{shortcut}(x, y). \tag{4}$$

Note that the simpler typecasting case discussed in Section 2.2 can in fact be understood as a very simple case of contraction and expansion.

The rule expressing a shortcut (i.e., contraction) cannot in general be represented in OWL, and this is well-known. In particular, if the graph representing the rule body is cyclic, this is not possible in many cases. Discussing this in detail is out of scope for this paper, but a detailed account of this can e.g. be found in [11,13].

However, let us work with the earlier example from rule (3), which is not cyclic. In this case we can convert the rule into OWL using rolification, which results in the following set of axioms.

$$\begin{gathered}
\text{Vessel} \equiv \exists R_{\text{Vessel}}.\text{Self}, \qquad \text{Cruise} \equiv \exists R_{\text{Cruise}}.\text{Self} \\
\text{Trajectory} \equiv \exists R_{\text{Trajectory}}.\text{Self}, \qquad \text{Segment} \equiv \exists R_{\text{Segment}}.\text{Self} \\
R_{\text{Segment}} \circ \text{hasSegment}^{-} \circ R_{\text{Trajectory}} \circ \\
\circ\, \text{hasTrajectory}^{-} \circ R_{\text{Cruise}} \circ \text{isUndertakenBy} \sqsubseteq \text{isTraversedBy}
\end{gathered}$$

The problem is again, of course, that the introduction of additional role chains may render the ontology to be outside OWL DL due to possible violations of regularity restrictions.

The cases where rules are cyclic pose particular challenges. We illustrate this by an example taken from [11]. The rule defines a shortcut indicating a review assignment with a conflict of interest.

$$\begin{aligned}
&\text{hasReviewAssignment}(v, x) \wedge \text{hasAuthor}(x, y) \wedge \text{atVenue} \wedge \\
&\wedge\, \text{hasSubmittedPaper}(v, u) \wedge \text{hasAuthor}(u, y) \wedge \text{atVenue}(u, z) \\
&\qquad\qquad \rightarrow \text{hasConflictingAssignedPaper}(v, x)
\end{aligned}$$

Approximate (sound but incomplete) translations of such cyclic rules into OWL are possible using DL-safe rules [9,15]. Better approximations (i.e., with not as much loss in terms of logical consequences) are possible using so-called *nominal schemas* [11,13]. While in the meantime some results have been obtained regarding efficient reasoning with nominal schemas [5,19], the topic does still require in depth exploration to obtain sufficient coverage for modeling purposes.

Assuming familiarity with rule to OWL conversion techniques as discussed e.g. in [11], we identify several research questions which address such conversion issues. Some of them have in fact already been exposed by our earlier discussions.

(1) Translation of rules usually requires rolification and the use of role chains, i.e. softening regularity restrictions would be extremely helpful.

(2) Approximate translation of rules (approximate in order to avoid regularity issues) currently requires the use of nominal schemas, for which efficient reasoning algorithms, as well as suitable modeling and reasoning tools, require further investigation.

(3) Translated rules often fall into the OWL EL fragment with additional use of inverse roles. While OWL EL requires regularity, the regularity requirement is not required for decidability of the logic. However, in the presence of inverse roles, together with a non-regular set of role chains, the logic becomes undecidable [1]. Softening the regularity requirement for OWL EL with addtional inverse roles would make it possible to translate more shortcut rules.

(4) Likewise, OWL EL (with the regularity restriction) together with inverse roles is no longer tractable. Research into conditions under which tractability is retained would be helpful in practice – see e.g. for [4] for some work related to this issue.

Another issue arising out of shortcuts is if ClassB, in our generic example, is actually a datatype, i.e., the infered role ‘shortcut’ shall be a datatype property.

Structurally, representation of the corresponding rule should follow the same method, however the resulting OWL axioms will then usually involve a role chain with a datatype property as the final, right-most role. However, the OWL standard currently does not allow this. We conjecture that allowing this would probably be a minor extension of the standard, but this still requires looking at.

3.2 Expansion

Expansion is the reverse of contraction, i.e. expanding from a shortcut into the graph, as in our generic example. It can be understood as a generalized version of the direction from Case 2a to 2b in Section 2.2 where a blank node is introduced, i.e. as a type of role introduction instead of using an elementary property.

Of course, simply reversing the implication arrow in rule (4) is insufficient, as quantification of the variables needs to be addressed. The appropriate axiomatization, in fact, is the following.

$$\begin{aligned}\text{shortcut}(x, y) \to \text{ClassA}(x) \land \text{ClassB}(y) \land \exists x_1 \ldots \exists x_n \exists y_1 \ldots \exists y_n (C_1(x_1) \land \ldots \\ \cdots \land C_n(x_n) \land R_1(y_1, y_2) \land \cdots \land R_k(y_k, y_{k+1}))\end{aligned}$$

Similar to the case above in Section 2.2, existential rules appear to be a suitable paradigm, in principle. However the potentially rather complex rule heads deserve considerable investigation, in particular if it is to be integrated with ontology reasoning.

A specific case which may also deserve studying is when the rule head may be translatable into a right-hand-side role chain, i.e. an axiom of the form

$$R \sqsubseteq R_1 \circ \cdots \circ R_n,$$

possible after some rolification. Right-hand-side role chains have been studied in the literature and in the general case they lead to undecidability, particularly when left-hand side role chains are present. Decidability by generalizing regularity restriction where shown by Mosurovic, et al [14]. On the other hand, the rule above can also be categorized into guarded TGDs [3] for which query answering is decidable. Note, however, that adding existential rules to OWL in general may cause the violation of guardedness condition, hence may not guarantee decidability.

4 Conclusions

We have seen that modeling issues arising in practice give rise to logical axioms which are currently not expressible within the OWL DL standard, and this prompts research questions which may ultimately lead to a suitable coverage in a later version of the standard. To provide an overview, we list the research questions raised by our discussion.

1. Relaxing RBox regularity constraints to make use of rolification easier, for several of the aspects mentioned above.

2. Relaxing RBox regularity constraints in the specific case of OWL EL with additional inverse roles would allow for the expression of more view contractions. Aspects to be considered would be both, decidability and tractability.
3. Develop more efficient reasoning algorithms and implementations for nominal schemas, as they are one way to circumvent the regularity issues arising from rolification.
4. Investigate reasoning aspects regarding role chains which end in datatype literals, including the issue of right-hand-side role chains.
5. Investigate right-hand-side role chains as a possible extension of OWL DL.
6. Investigate the integration of existential rules with OWL DL, in particular for complex rule heads.

Acknowledgements. This work was supported by the National Science Foundation under award 1017225 *III: Small: TROn – Tractable Reasoning with Ontologies* and award 1440202 *EarthCube Building Blocks: Collaborative Proposal: GeoLink – Leveraging Semantics and Linked Data for Data Sharing and Discovery in the Geosciences.*

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Kaelbling, L.P., Saffiotti, A. (eds.) IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005. pp. 364–369 (2005)
2. Beeri, C., Vardi, M.Y.: A proof procedure for data dependencies. Journal of the ACM 31(4), 718–741 (1984)
3. Calì, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. J. Artif. Intell. Res. (JAIR) 48, 115–174 (2013)
4. Carral, D., Feier, C., Grau, B.C., Hitzler, P., Horrocks, I.: *EL*-ifying ontologies. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) Automated Reasoning – 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8562, pp. 464–479. Springer (2014)
5. Carral, D., Wang, C., Hitzler, P.: Towards an efficient algorithm to reason over description logics extended with nominal schemas. In: Faber, W., Lembo, D. (eds.) Web Reasoning and Rule Systems – 7th International Conference, RR 2013, Mannheim, Germany, July 27-29, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7994, pp. 65–79. Springer (2013)
6. Gangemi, A.: Ontology design patterns for semantic web content. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) The Semantic Web – ISWC 2005, 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3729, pp. 262–276. Springer (2005)
7. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S. (eds.): OWL 2 Web Ontology Language: Primer. W3C Proposed Recommendation 22 September 2009 (2009), available from http://www.w3.org/TR/owl2-primer/
8. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. CRC Press/Chapman & Hall (2010)

9. Hitzler, P., Parsia, B.: Ontologies and rules. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 111–132. Springer (2009)
10. Krisnadhi, A., Hu, Y., Janowicz, K., Hitzler, P., Arko, R.A., Carbotte, S., Chandler, C., Cheatham, M., Fils, D., Finin, T.W., Ji, P., Jones, M.B., Karima, N., Lehnert, K., Mickle, A., Narock, T.W., O'Brien, M., Raymond, L., Shepherd, A., Schildhauer, M., Wiebe, P.: The GeoLink modular oceanography ontology. In: Arenas, M., Corcho, Ó., Simperl, E., Strohmaier, M., d'Aquin, M., Srinivas, K., Groth, P.T., Dumontier, M., Heflin, J., Thirunarayan, K., Staab, S. (eds.) The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9367, pp. 301–309. Springer (2015)
11. Krisnadhi, A., Maier, F., Hitzler, P.: OWL and rules. In: Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P.F. (eds.) Reasoning Web. Semantic Technologies for the Web of Data – 7th International Summer School 2011, Galway, Ireland, August 23-27, 2011, Tutorial Lectures. Lecture Notes in Computer Science, vol. 6848, pp. 382–415. Springer (2011)
12. Krisnadhi, A.A., Hu, Y., Janowicz, K., Hitzler, P., Arko, R., Carbotte, S., Chandler, C., Cheatham, M., Fils, D., Finin, T., Ji, P., Jones, M., Karima, N., Lehnert, K., Mickle, A., Narock, T., O'Brien, M., Raymond, L., Shepherd, A., Schildhauer, M., Wiebe, P.: The GeoLink framework for pattern-based Linked Data integration. In: Proceedings of the ISWC 2015 Posters & Demonstrations Track a track within the 14th International Semantic Web Conference, ISWC 2015, Bethlehem, PA, USA, October 13, 2015 (2015)
13. Krötzsch, M., Maier, F., Krisnadhi, A., Hitzler, P.: A better uncle for OWL: nominal schemas for integrating rules and ontologies. In: Srinivasan, S., Ramamritham, K., Kumar, A., Ravindra, M.P., Bertino, E., Kumar, R. (eds.) Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 – April 1, 2011. pp. 645–654. ACM (2011)
14. Mosurovic, N., Krdzavac, H., Graves, M., Zakharyaschev, M.: A decidable extension of SROIQ with complex role chains and unions. Journal of Artificial Intelligence Research 47, 809–851 (2013)
15. Motik, B., Sattler, U., Studer, R.: Query answering for OWL-DL with rules. Journal of Web Semantics 3(1), 41–60 (2005)
16. Noy, N. (ed.): Representing Classes As Property Values on the Semantic Web. W3C Working Group Note (5 April 2005), `http://www.w3.org/TR/swbp-classes-as-values/`
17. Rodríguez-Doncel, V., Krisnadhi, A.A., Hitzler, P., Cheatham, M., Karima, N., Amini, R.: Pattern-based linked data publication: The linked chess dataset case. In: Hartig, O., Sequeda, J., Hogan, A. (eds.) Proceedings of the 6th International Workshop on Consuming Linked Data co-located with 14th International Semantic Web Conference (ISWC 2105), Bethlehem, Pennsylvania, US, October 12th, 2015. CEUR Workshop Proceedings, vol. 1426. CEUR-WS.org (2015)
18. Rudolph, S., Krötzsch, M., Hitzler, P.: All elephants are bigger than all mice. In: Baader, F., Lutz, C., Motik, B. (eds.) Proceedings of the 21st International Workshop on Description Logics (DL2008), Dresden, Germany, May 13-16, 2008. CEUR Workshop Proceedings, vol. 353. CEUR-WS.org (2008)
19. Steigmiller, A., Glimm, B., Liebig, T.: Reasoning with nominal schemas through absorption. Journal of Automated Reasoning 53(4), 351–405 (2014)