Modeling in OWL 2 without Restrictions

Michael Schneider[1], Sebastian Rudolph[2], and Geoff Sutcliffe[3]

[1] FZI Research Center for Information Technology, Germany
[2] Institute AIFB, Karlsruhe Institute of Technology, Germany
[3] University of Miami, USA

Abstract. The Semantic Web ontology language OWL 2 DL comes with a variety of language features that enable sophisticated and practically useful modeling. However, the use of these features has been severely restricted in order to retain decidability of the language. For example, OWL 2 DL does not allow a property to be both transitive and asymmetric, which would be desirable, e.g., for representing an ancestor relation. In this paper, we argue that the so-called "global restrictions" of OWL 2 DL preclude many useful forms of modeling, by providing a catalog of basic modeling patterns that would be available in OWL 2 DL if the global restrictions were discarded. We then report on the results of evaluating several state-of-the-art OWL 2 DL reasoners on problems that use combinations of features in a way that the global restrictions are violated. The systems turn out to rely heavily on the global restrictions and are thus largely incapable of coping with the modeling patterns. Based on our observations, we make suggestions for future lines of research on expressive description logic-style OWL reasoning.

Keywords: Semantic Web, Ontology, Modeling, OWL DL

1 Introduction

The Semantic Web ontology language OWL 2 DL [9,4,3] was standardized by the World Wide Web Consortium (W3C) in 2009 (and updated in 2012) as a description logic-style formalism of high expressivity that still guarantees algorithmic decidability of core reasoning tasks such as ontology satisfiability and entailment checking. The language comes with a large number of language features that enable sophisticated modeling in many application domains. However, the use of these features has been restricted in a variety of ways in order to retain decidability of the language. For instance, via a collection of *"global restrictions"*, particular uses of certain features or a combination of these features are explicitly constrained. A class of features for which a large number of global restrictions have been defined are the so-called *"complex properties"*, that is, transitive properties and properties defined through property chain axioms. For example, OWL 2 DL allows for declaring transitive properties as well as asymmetric properties, but does not allow a property to be both transitive and asymmetric, as would be natural for a property representing an ancestor relation or, more generally, any strict partial order. In this paper we analyze some of the practical ramifications of the global restrictions on complex properties. We argue that dropping the global restrictions would result in a large number of additional useful and relevant modeling options for knowledge representation with OWL 2. We believe that in many practical cases these advantages outweigh the theoretical advantages of decidability.

We start in Section 2 with a concise overview of OWL 2 DL and its global syntactic restrictions. In Section 3 we provide a catalog of basic modeling patterns that use complex properties in natural ways, but which are disallowed by the global restrictions. For all patterns, we give example use cases supporting their usefulness and practical relevance. For ontology authors, the catalog offers a large set of new useful modeling options and demonstrates the extended modeling potential available, in principle at least, through the OWL 2 standard. In the same way, the catalog allows for a better understanding of the actual limitations of modeling in OWL 2 DL due to the global restrictions. To our knowledge, no comparable catalog of patterns exists.

In Section 4 we report on the results of an evaluation of several state-of-the-art OWL 2 DL reasoners using test problems that are based on the modeling patterns. The evaluation results provide an understanding of what can be expected from existing reasoners when they are applied to input data that violates the global restrictions of OWL 2 DL. It has to be pointed out that such an investigation is meaningful, as the OWL 2 standard does *not* require OWL 2 DL reasoners to reject input beyond OWL 2 DL, but only specifies how such tools have to behave on legal OWL 2 DL input (see the definition of tool conformance for OWL 2 DL entailment checkers in [8]). It has already been noted that OWL 2 DL reasoners can frequently be applied to input data that is significantly beyond OWL 2 DL without producing processing errors, and that they sometimes produce the expected results [7]. Compared to that study, the test problems used in this paper intuitively appear to be more digestible to OWL 2 DL reasoners. They can be expressed in the OWL 2 structural specification [4] (of which OWL 2 DL is a syntactic fragment), which has a precise meaning under the OWL 2 direct semantics [3] – the semantics underlying OWL 2 DL. This gave rise to a hope that existing OWL 2 DL reasoners might be able to cope with our modeling patterns. However, the evaluation reveals that all the OWL 2 DL reasoners failed on all the modeling patterns.

A technical report is available [6] and includes as a possible path forward an analysis of the the practical feasibility of using first-order logic (FOL) reasoning technology [5] to reason in OWL 2 DL without the global restrictions, and gives suggestions for future lines of research on expressive description logic-style OWL reasoning. The technical report further includes appendices with additional technical information on which the paper is based. This information is also available in the *supplementary material* at `http://www.fzi.de/downloads/ipe/schneid/srs12-complexrelations.zip` .

2 OWL 2 DL and Global Restrictions

2.1 OWL 2 DL

For space reasons, we refrain from repeating the structural specification of OWL 2 DL, and instead refer the reader to [4] for the complete details. Here we focus on the aspects important for our argument.

Recall that the basic modeling primitives in OWL are individuals, classes and properties, where the latter are interpreted by binary relations and strictly subdivided into data properties and object properties. Compared to its predecessor OWL 1, OWL 2 has been significantly extended by ways to describe characteristics and interdependencies on the object property level. In particular, `SubObjectPropertyOf` statements are allowed to take *property chains* as their first argument, as, e.g., in

```
SubObjectPropertyOf( ObjectPropertyChain( :hasParent :hasBrother ) :hasUncle )
```

expressing that somebody's parent's brother is the uncle of that somebody. In database terms one could say that the uncle relation subsumes the join of the parent relation with the brother relation. Another novel property-centric modeling feature is *property disjointness*, as, e.g., in

```
DisjointObjectProperties( :hasParent :hasUncle )
```

expressing that somebody's parent cannot be that somebody's uncle. Furthermore, OWL 2 allows for characterizing properties as functional, inverse-functional, reflexive, irreflexive, symmetric, asymmetric, or transitive as, e.g., in

```
TransitiveObjectProperty( :hasAncestor )
```

Recall further that the semantics of OWL 2 DL, called *direct semantics* [3] is established along the typical model-theoretic semantics in description logics [1], and is well-defined for any structurally specified OWL ontology even if it violates the global restrictions.

2.2 Global Restrictions

In order to ensure decidability despite the high expressivity of the diverse modeling features in OWL, the ways in which these features are allowed to interact had to be restricted. This led to the so-called *global restrictions* that an OWL 2 DL ontology has to satisfy (see Chapter 11 of [4]). The name "global restrictions" comes from the fact that satisfaction of these restrictions cannot be decided by looking at the ontology axioms in isolation but it depends on their interplay.

At the core of the restrictions is the notion of *simple* versus *complex* object properties. Roughly speaking, an object property is called complex, if it can be inferred from the join of two or more other object properties. For example, the above subproperty axiom renders the uncle property complex. The same holds for the ancestor property, since transitivity of a relation essentially means that the relation subsumes its own self-join.

The global restrictions severely constrain the ways in which complex properties can be used: according to the *restriction on simple properties*, complex properties are not allowed to occur in cardinality restrictions, self-restrictions, and property disjointness statements, nor is a complex property allowed to be characterized as functional, inverse-functional, irreflexive, or asymmetric.

Another severe restriction is on the co-occurrence of subproperty axioms, that is, the *restriction on the property hierarchy.* The rationale behind this rather technical restriction is to ensure that the set of property chains used to infer a complex property can be described as a regular language. Next to discarding certain subproperty axioms right away, this also prohibits the coexistence of certain such axioms.

Further restrictions apply to OWL 2 DL (referring to the use of blank nodes), but they are not of interest in this paper.

3 Modeling Patterns

This section presents a catalog of modeling patterns based on usage of OWL 2 language features in a way that violates the global restrictions of OWL 2 DL. The catalog consists of twelve modeling patterns, most of them representing different combinations of axioms defining complex properties, such as transitivity axioms, and language constructs that may only be used with simple properties, such as asymmetric property axioms; see Section 2 for a more detailed list of disallowed combinations of language constructs in OWL 2 DL. Each modeling pattern is described with a concrete example of a family relationship given in OWL 2 functional syntax, and is accompanied by an explanation of the conflicts with the OWL 2 DL specification. Additional use cases from other application areas provide evidence for the generality, usefulness, and relevance of the pattern.

We have to point out that the patterns were *not* taken from any existing OWL ontologies. As the patterns are explicitly disallowed in OWL 2 DL, and as many of the involved language features were introduced only recently as part of OWL 2, one cannot expect to find many of these patterns in real-world ontologies today. Rather, the goal is to demonstrate the drastic increase of modeling power in case the global restrictions of OWL 2 DL are discarded. Our motivations for choosing the modeling patterns were simplicity, plausibility, potential relevance, and generality.

We are aware that for some of the patterns it is possible to find a semantically equivalent reformulation that is valid in OWL 2 DL. However, the purpose of our pattern catalog is *not* to present semantic scenarios that cannot be expressed in OWL 2 DL, but rather to offer to ontology authors a set of new modeling options that appear natural and simple using the features of OWL 2. We believe that for an ontology author, a complex or non-obvious reformulation of a pattern, in order to keep the ontology in OWL 2 DL, will often be unacceptable. Still, we consider work on such translations relevant as a means of "repairing" ontologies that use our modeling patterns, so that OWL 2 DL reasoners have a better chance of coping with such input (cf. the results in Section 4).

3.1 Strict Partial Orders

Strict partial orders are asymmetric transitive relations, such as the ancestor relationship between people:

```
TransitiveObjectProperty( :hasAncestor )
AsymmetricObjectProperty( :hasAncestor )
```

OWL 2 DL does not allow transitive properties to be asymmetric. Additional use cases include: comparison relations such as *greater-than*, part-whole relationships, and operational research tasks such as critical path analysis and supply chain management.

3.2 Characterized Composite Relations

Property chain axioms allow composite relations to be built, such as the uncle relation in terms of the parent and brother relations. Naturally, the uncle relation should be specified to be asymmetric:

```
SubObjectPropertyOf(
  ObjectPropertyChain( :hasParent :hasBrother )
  :hasUncle )
AsymmetricObjectProperty( :hasUncle )
```

OWL 2 DL does not allow composite properties to be asymmetric. Another use case is an asymmetric nth-order predecessor relation for a fixed number n, such as a grandparent defined as a parent's parent ($n = 2$).

3.3 Disjoint Transitive Relation Pairs

Relations are often defined as pairs of complementary but mutually exclusive terms that are transitive, such as the ancestor and descendant relationships:

```
TransitiveObjectProperty( :hasAncestor )
TransitiveObjectProperty( :hasDescendant )
DisjointObjectProperties( :hasDescendant :hasAncestor )
```

OWL 2 DL does not allow disjointness of transitive properties. Another use case is disjoint pairs of comparison relations, such as *greater-than* and *smaller-than*. Another disallowed example of two disjoint relations of which only one is transitive is given by the SKOS semantic relations `skos:broaderTransitive` and `skos:related` [2] (S24, S27).

3.4 Disjoint Composite Relations

Relations composed using property chain axioms are often disjoint from one or more of the component relations. For example, when composing the uncle relation in terms of the parent and brother relations, then, realistically, all three relations are mutually disjoint:

```
SubObjectPropertyOf(
  ObjectPropertyChain( :hasParent :hasBrother )
  :hasUncle )
DisjointObjectProperties( :hasUncle :hasParent :hasBrother )
```

OWL 2 DL does not allow disjointness of composite properties. Another use case is an nth-order predecessor relation for a fixed number n, such as a grandparent defined as a parent's parent ($n = 2$), where the grandparent and parent relations are disjoint.

3.5 Lower-Bounded Transitive Relations

For some transitive relations it may be desirable to specify the minimum number of relationships per individual. For example, every person has at least two ancestors:

```
TransitiveObjectProperty( :hasAncestor )
SubClassOf(
  :Person
  ObjectMinCardinality( 2 :hasAncestor :Person ) )
```

OWL 2 DL does not allow cardinality restrictions on transitive properties. Other use cases are comparison relations over unbounded domains, such as *greater-than* for numbers, where for any given number n there are always numbers $m > n$.

3.6 Functional Composite Relations

For some relations composed by property chain axioms it may be desirable to define them to be functional. For example, every person has at most one living maternal grandfather, being the father of the person's mother:

```
SubObjectPropertyOf(
  ObjectPropertyChain( :hasMother :hasFather )
  :hasMaternalGrandfather )
FunctionalObjectProperty( :hasMaternalGrandfather )
```

OWL 2 DL does not allow composite properties to be functional. An additional use case is a part-ownership relation, in scenarios where items can only have a single owner and where the owner of an item also owns all parts of the item.

3.7 Propagated Relations

Some relationships between two individuals may "propagate" to two other individuals, due to a specific constellation of relationships that holds between all four individuals. For example, if Mary has mother Susan, and Bill has father John, where Susan and John are relatives, then Mary and Bill are also relatives. This can be expressed using property chain axioms:

```
SubObjectPropertyOf(
  ObjectPropertyChain(
    :hasMother
    :hasRelative
    ObjectInverseOf( :hasFather ) )
  :hasRelative )
```

This representation violates the regularity conditions for the property hierarchy of OWL 2 DL, as in chains of size 3 or larger, an inner property of the chain (`:hasRelative` in position 2) must not also occur as the composite property. An additional use case would be to characterize identical composite items, such as computers, to have identical corresponding components, such as the computer's processors.

3.8 Interlaced Relation Definitions

Although there is no general method in OWL 2 to fully define a composite relation, one can sometimes encode a close characterization by interlacing two property chain axioms. For example, one can define an uncle as a cousin's father, and a cousin as an uncle's child:

```
SubObjectPropertyOf(
  ObjectPropertyChain( :hasCousin :hasFather )
  :hasUncle )
SubObjectPropertyOf(
  ObjectPropertyChain( :hasUncle ObjectInverseOf( :hasFather ) )
  :hasCousin )
```

Circular dependencies on the property hierarchy violate the regularity conditions of OWL 2 DL.

3.9 Scoped Equivalence Relations

Equivalence relations are transitive, symmetric and reflexive, but for reflexivity, a global scope is often not desirable, as it would entail that *everything* has the relationship. For example, being a relative to someone may be seen as an equivalence relation, provided that one accepts that everyone is a relative of himself. However, one would probably want to limit this relation to people, excluding, for instance, machines or ideas. OWL 2 supports this notion of a "locally-reflexive" property by self-restrictions:

```
SymmetricObjectProperty( :hasRelativeOrSelf )
TransitiveObjectProperty( :hasRelativeOrSelf )
EquivalentClasses( :Person ObjectHasSelf( :hasRelativeOrSelf ) )
```

OWL 2 DL does not allow self-restriction of transitive properties. Other use cases include the grouping of people according to some feature, such as having the same profession or nationality. SKOS specifies the mapping property `skos:exactMatch` as symmetric and transitive [2] (S44, S45), and it would be plausible and consistent with the SKOS standard to additionally make it locally reflexive to the class of SKOS concepts.

3.10 Quasi-Reflexive-Transitive Closures

The reflexive-transitive closure of a parent relation defined over the class of people is the smallest super relation that is both transitive and reflexive, where reflexivity is scoped to the class of people, i.e., a person's ancestor or oneself. While it is not possible to represent the smallest such relation in OWL 2, a coarse approximation is possible using a self-restricted transitive super property:

```
SubObjectPropertyOf( :hasParent :hasAncestorOrSelf )
TransitiveObjectProperty( :hasAncestorOrSelf )
EquivalentClasses( :Person ObjectHasSelf( :hasAncestorOrSelf ) )
```

OWL 2 DL does not allow self-restriction of transitive properties. Another example is `skos:broaderTransitive`, the transitive extension of the SKOS semantic property `skos:broader` [2, S22,S24], for which it would be plausible and consistent with the SKOS standard to additionally make it locally reflexive to the class of SKOS concepts.

3.11 Homocyclic Relationships

Cyclic relationships constructed from one binary relation, such as the "loves" relation, may be of arbitrary size. For example, a person may love only himself or another person mutually, or there may be a cycle of unreturned love including several people. Each person in such a cycle can be represented as an instance of the class of "loved lovers" and, thus, instanceship in this class indicates that a person is part of such a cyclic relationship. Class instanceship can be expressed in terms of a self-restricted transitive super property of the loves property:

```
SubObjectPropertyOf( :loves :z )
TransitiveObjectProperty( :z )
SubClassOf( ObjectHasSelf( :z ) :LovedLover )
```

OWL 2 DL does not allow self-restriction of transitive properties. Another use case is chemical ring molecules of arbitrary size, where all bonds are of the same sort, such as Cycloalkanes.

3.12 Heterocyclic Relationships

Certain cyclic relations or coincidence relations can be composed from a set of different basic relations, such as the concept of a legitimate child, that is, a person with a father and a mother who are married. Occurrence of such relationships in a knowledge base can be indicated by instanceship in a class of legitimate children, modeled using a property chain axiom and a self-restriction:

```
SubObjectPropertyOf(
  ObjectPropertyChain(
    :hasMother
    :hasSpouse
    ObjectInverseOf( :hasFather ) )
  :z )
SubClassOf( ObjectHasSelf( :z ) :LegitimateChild )
```

OWL 2 DL does not allow self-restriction of composite properties. Another use case is circular molecules of a fixed size that are built from different sorts of bonds, such as Furan.

4 Evaluation of State-of-the-Art Semantic Web Reasoners

We now report on the results of evaluating several state-of-the-art OWL 2 DL reasoners using test problems based on the modeling patterns of Section 3. The focus was on finding out whether or not the reasoners can cope with the modeling patterns. As mentioned in Section 1, compliant OWL 2 DL reasoners are *not* required to reject input beyond the specification of OWL 2 DL, and experience shows that existing systems often do reason upon such input. Hence it is a legitimate question to ask how they behave on our modeling patterns. To give an answer, we checked whether the reasoners are able to recognize certain "obvious looking" logical conclusions from the modeling patterns according to the OWL 2 direct semantics. Reasoning performance was *not* considered an important aspect of our evaluation.

Test Data. We created a test suite consisting of one test case per modeling pattern. Each test case is built from two small ontologies, a premise and a conjecture. The premise covers the main example of the corresponding modeling pattern given in Section 3, typically extended by some additional assertions. The conjecture is a small set of assertions that follow logically from the premise. Both the premise and the conjecture ontology conform syntactically to the OWL 2 structural specification, and the conjecture is entailed from the premise according to the OWL 2 direct semantics. The test cases were designed to be "not too difficult to solve", so that the OWL 2 DL reasoners do not fail due to high reasoning complexity. The complete test suite is described in full detail in the appendix of the technical report [6], and in the supplementary material.

Reasoners. We selected currently available reasoners that represent the state of the art in OWL 2 DL reasoning. We used OWL API 3.2.4 for parsing the test cases, and all reasoners were accessed via their respective OWL API reasoner interfaces.[4]

– **FaCT++ 1.5.3** (`http://owl.man.ac.uk/factplusplus`), created at the University of Manchester, England, is a tableaux-based OWL 2 DL reasoner.
– **HermiT 1.3.6** (`http://hermit-reasoner.com`), created at the University of Oxford, England, is a tableaux-based OWL 2 DL reasoner.
– **Pellet 2.3.0** (`http://clarkparsia.com/pellet`), created by Clark & Parsia, USA, is a tableaux-based OWL 2 DL reasoner.

Testing Environment. All tests were conducted on a mobile computer "Lenovo ThinkPad T410s" with an Intel® Core™ i5 M520 CPU (4 cores) at 2.4 GHz speed, 4 GB RAM, with Microsoft Windows 7 Professional (64-Bit) as the operating system. The CPU timeout for applying a reasoner to a test case was 300 seconds. The possible outcomes of the test runs are as follows:

– '+': termination with correct result
– '−': termination with wrong result
– '?': processing error or timeout

Results. Table 1 shows the outcomes of the evaluation. The details of the results can be found in the appendix of the technical report [6] and in the supplementary material. In summary, all reasoners failed on all test cases. *FaCT++* signalled errors on all test cases with error messages that in most cases correctly indicated which global restriction was violated. *HermiT* signaled ten errors with error messages that correctly identified the global restriction that was violated, while two test cases were wrongly recognized as non-entailments. *Pellet* signaled an error in only one case, and wrongly recognized all other test cases as non-entailments. In the majority of cases a warning message was found in the log file, which explained that Pellet had recognized a violated global restriction and

[4] OWL API homepage: `http://owlapi.sourceforge.net`

	01	***02***	***03***	***04***	***05***	***06***	***07***	***08***	***09***	***10***	***11***	***12***
Fact++	?	?	?	?	?	?	?	?	?	?	?	?
HermiT	?	?	?	?	?	?	?	?	?	?	–	–
Pellet	–	–	–	–	–	–	–	–	–	–	?	–

Table 1. Entailment checking results for the OWL 2 DL reasoners using the twelve entailments of the "Complex Family Relations" test suite.

chosen to ignore one or more of the premise axioms as a way to resolve the conflict. Without those axioms it was not possible to infer the conclusion ontologies. The CPU times taken by all the reasoners were below 20 ms for the majority of test cases, and FaCT++ always returned after less than 10 ms. In order to find out whether the bad outcomes were at least partially due to the use of the OWL API, we compared the logical axioms and declarations after parsing the test case data with those in the original test cases, and found no differences. This indicates that the reasoners are mainly responsible for the outcomes themselves.

Discussion. These results strongly indicate that today's OWL 2 DL reasoners cannot reliably be used with input that violates the global restrictions of OWL 2 DL. Note that this does not mean to accuse these reasoners of malfunctioning, they just do not go the extra mile beyond their specified input language and hence are not suitable for reasoning in the extended language that we are targeting.[5] Apparently two different strategies are used by the reasoners in this situation: FaCT++ and HermiT rigidly reject the input, while Pellet processes the input with some of the conflicting axioms being ignored, which may lead to missing or wrong results. For users who want to apply some of the modeling patterns introduced earlier in this paper, none of these strategies is acceptable. Therefore, a different strategy that does not have these problems is needed. We propose and evaluate one such strategy in the technical report [6].

5 Conclusion

In this paper we have shown that many useful and relevant modeling options were available in OWL 2 DL if the global restrictions on complex properties would be relinquished. We have presented a catalog of twelve basic useful modeling patterns that are in the scope of the unrestricted structural specification and the direct semantics of OWL 2 but are beyond the scope of OWL 2 DL, including strict partial orders and different forms of circular relationships. Although the

[5] For someone familiar with the methods used in state-of-the-art OWL reasoners this fact does not come as a big surprise. For instance, the restriction on the property hierarchy is a crucial prerequisite for preprocessing the ontology ahead of the actual core reasoning procedures.

OWL 2 DL standard does not prevent compliant reasoners from processing such input, all the state-of-the-art OWL 2 DL reasoners that we tested were unable to cope with these modeling scenarios.

In our technical report [6] we have analysed the use of generic FOL reasoning technology for reasoning in the unrestricted OWL 2 direct semantics, and our experiments led to fully satisfying results on our test data. We therefore suggest building loosely coupled hybrid OWL 2 reasoners from traditional tableaux-based systems and generic FOL systems, to cover a wider range of input without sacrificing the completeness guarantees and high efficiency of today's OWL 2 DL reasoners on legal OWL 2 DL input. As further research tasks we propose investigating the optimization potential of FOL reasoning for unrestricted OWL, and determining the most relevant use cases of non-finite model finding for OWL reasoning, and the development of specialized reasoning methods for them.

Acknowledgements. Michael Schneider has been supported by the projects *SEALS* (European Commission, EU-IST-2009-238975) and *THESEUS* (German Federal Ministry of Economics and Technology, FK OIMQ07019). Sebastian Rudolph is supported by the project ExpresST funded by the German Research Foundation (DFG).

References

1. Baader, F., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
2. Miles, A., Bechhofer, S. (eds.): SKOS Simple Knowledge Organization System Reference. W3C Recommendation (18 August 2009), access: `http://www.w3.org/TR/skos-reference/`
3. Motik, B., Patel-Schneider, P.F., Grau, B.C. (eds.): OWL 2 Web Ontology Language: Direct Semantics (Second Edition). W3C Recommendation (11 December 2012), access: `http://www.w3.org/TR/owl2-direct-semantics/`
4. Motik, B., Patel-Schneider, P.F., Parsia, B. (eds.): OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax (Second Edition). W3C Recommendation (11 December 2012), access: `http://www.w3.org/TR/owl2-syntax/`
5. Robinson, A., Voronkov, A.: Handbook of Automated Reasoning. Elsevier Science (2001)
6. Schneider, M., Rudolph, S., Sutcliffe, G.: Modeling in OWL 2 without Restrictions. Extended technical report, FZI Research Center for Information Technology (2012), available at: `http://arxiv.org/abs/1212.2902`
7. Schneider, M., Sutcliffe, G.: Reasoning in the OWL 2 Full Ontology Language using First-Order Automated Theorem Proving. In: Proc. CADE 23. LNAI, vol. 6803, pp. 446–460 (2011), extended version at `http://arxiv.org/abs/1108.0155`
8. Smith, M., Horrocks, I., Krötzsch, M., Glimm, B. (eds.): OWL 2 Web Ontology Language: Conformance (Second Edition). W3C Recommendation (11 December 2012), access: `http://www.w3.org/TR/owl2-conformance/`
9. W3C OWL Working Group (ed.): OWL 2 Web Ontology Language: Document Overview (Second Edition). W3C Recommendation (11 December 2012), access: `http://www.w3.org/TR/owl2-overview/`