Comparing OWL Semantics

David Turner, Jeremy J. Carroll
Digital Media Systems Laboratory
HP Laboratories Bristol
HPL-2007-146
September 4, 2007*

OWL, semantic web

The OWL Web Ontology Language is endowed with two model theories, reflecting its origins as a compromise between two different communities. By design these model theories give rise to very similar semantics, and a precise statement of the correspondence between the model theories is conjectured with a sketch proof at the end of the OWL semantics specification document. We have filled in the details of this sketch proof using the Isabelle/HOL proof assistant, and developed machinery for further study of the formal semantics of OWL. Our study was sufficiently detailed to find a handful of minor errors in the specification of the semantics of OWL that previous work had overlooked. We also sought a stronger result by showing a partial converse to the known correspondence, but it proved impossible to achieve this within our time constraints; instead we conjecture a possible method for strengthening the correspondence.

* Internal Accession Date Only

Approved for External Publication

Comparing OWL Semantics

Dave Turner and Jeremy J. Carroll

HP Laboratories, Bristol, UK

Abstract. The OWL Web Ontology Language is endowed with two model theories, reflecting its origins as a compromise between two different communities. By design these model theories give rise to very similar semantics, and a precise statement of the correspondence between the model theories is conjectured with a sketch proof at the end of the OWL semantics specification document. We have filled in the details of this sketch proof using the Isabelle/HOL proof assistant, and developed machinery for further study of the formal semantics of OWL. Our study was sufficiently detailed to find a handful of minor errors in the specification of the semantics of OWL that previous work had overlooked. We also sought a stronger result by showing a partial converse to the known correspondence, but it proved impossible to achieve this within our time constraints; instead we conjecture a possible method for strengthening the correspondence.

1 Introduction

The two foundational recommendations of the Semantic Web are the Resource Description Framework (RDF)[1] and the Web Ontology Language (OWL)[2]. RDF is based on the earlier RDF Model and Syntax recommendation[3] while OWL is based on the DAML+OIL ontology language[4], which in turn was derived from more than a decade of research into description logics, e.g. [5].

The two recommendations were developed concurrently, with good communication between the two groups, and indeed several participants were in both groups. However, there were some key differences in world-view caused by the backgrounds of the specifications.

RDF is intended to provide a simple and uniform method for describing "things" (or "resources" in RDF parlance). It is intended to be Web-friendly, following the mantra that "anyone can say anything about anything"[6]. When taken to an extreme this idea can give rise to paradoxes and, in some sense, was dropped during the standardisation process, but it remains a powerful statement of the vision behind RDF. One of its consequences is that the concepts that can be used to make statements about resources are themselves describable in the same way: aspects which a traditional logic would view as part of the metalanguage are instead part of the object language.

The Description Logic (DL) community did not take this approach, and instead developed sophisticated syntaxes for describing resources following the usual approach that the primitives of the language are not themselves available

as the objects of discussion. Such syntaxes also divide the universe of discourse into disjoint categories of individuals, classes and properties, and restrict the permissible statements to be well-typed with respect to these categories. From the point of view of the RDF mantra, these are unwarranted limitations on the descriptive power of the language. For example, in RDF everything has type `rdf:Resource`, including the concept `rdf:Resource` itself, but a DL would view such a statement as ill-typed and would forbid it.

Both RDF and OWL are equipped with formal semantics, described model-theoretically[7, 8]. The resolution of the conceptual mismatch between the two languages is made principally in the OWL semantics specification, although historically the two languages were developed roughly in parallel. We sketch the resolution in the next section. For the purposes of our introduction it suffices to note the following issues with the resolution:

1. Its details are very complicated.
2. The proofs included in the specifications skip over certain details to reduce their complexity and aid their presentation.
3. The main result is not as strong as one might have hoped: only one direction of implication holds.

We believe that the complexity is largely unavoidable. We note that one of the drivers behind the current work on OWL 1.1[9] is to increase expressivity which will inevitably increase complexity. Our work aimed to produce more compelling proofs of the main results by making use of Isabelle/HOL, a computer-assisted higher-order logic (HOL) proof tool, to ensure that we had considered all the details missed in a hand proof. We briefly describe this approach in section 3, and the full source code of our proofs is included as a Web appendix to this report which may be found at the following address.

`http://www.hpl.hp.com/techreports/2007/HPL-2007-146/sourcecode`

The contribution of this paper is to present a detailed method for the proof of Theorem 2 of the OWL semantics specification, which asserts a correspondence between the semantics of OWL viewed as a DL and of OWL viewed as an extension of RDF. We begin with definitions in section 4, state the main theorems and lemmas in section 6, and discuss the proofs themselves in section 7. As a side effect of our detailed study we found a number of minor defects in the OWL and RDF semantics specifications which we outline in section 5.

We also present our efforts towards showing a partial converse to this theorem in section 8. These efforts were unsuccessful because of time constraints, but we discuss the main difficulties that we encountered and some possible methods for overcoming them.

2 Outline of OWL DL/OWL Full Resolution

The resolution of the conceptual mismatch between RDF and DL was decided by the Web Ontology Working Group as part of their issue 5.3[10].

The basic idea is that OWL has two syntaxes: the OWL Abstract Syntax is a traditional abstract syntax tree (AST) approach, and the RDF Graph syntax is a collection of RDF triples.

Section 4 of the OWL Semantics and Abstract Syntax (S&AS)[8] specification defines a relation T between these two syntaxes in the form of a collection of nondeterministic mapping rules. Not every RDF graph has a related abstract syntax representation. There are then two main strands of model theory: the Direct Semantics (OWL S&AS section 3) gives a semantics to OWL ontologies represented as ASTs, and the RDFS-compatible Semantics (OWL S&AS section 5) gives a semantics to OWL ontologies represented as RDF graphs.

The intent of the resolution was that there is a strong correspondence between the entailments sanctioned by the two semantics for ontologies that correspond under the relation T. Originally it was intended that this strong correspondence should be an "if-and-only-if" statement: If a' and b' are respective translations of a and b under T, then a entails b if and only if a' entails b'.

Unfortunately, it proved too difficult for the working group to arrange for this correspondence to hold in the time available, so the final resolution was to weaken it to a single implication: If a' and b' are RDF graphs which are respective translations of the ASTs a and b under T, then a entails b implies that a' entails b'. Indeed, the OWL Test Cases includes a number of tests[11–14] to demonstrate that the converse is indeed false.

3 Using Isabelle/HOL

Due to the complexity of both OWL and RDF, we found it implausible that a hand-proof of this correspondence would not contain errors, and without being able to identify those errors it is difficult to assess their importance. Therefore we decided to use the proof tool Isabelle/HOL to ensure that we had checked all the necessary details. The philosophical concerns with computer-proof over human-proof are mostly outside the scope of this paper, and we take a very pragmatic viewpoint: our computer proof found genuine errors that previous human proofs had missed, and demonstrates a method that a human could also follow, so we consider it to be a reasonably sound method.

The language of Isabelle/HOL is a strongly-typed predicate calculus, similar to the usual metalanguage of mathematicians. Furthermore, it has powerful tools for constructing definitions by structural recursion and performing proofs by structural induction, which makes it well suited for studying OWL from its Abstract Syntax viewpoint. The input for Isabelle consists of "proof scripts" which contain definitions and statements of lemmas, and descriptions of the steps required to construct a proof. These descriptions are written in the high-level language *Isar*, and resemble the way that mathematicians describe proofs on paper: with practice it is straightforward to translate a sufficiently detailed hand-proof into Isar. Isabelle may check a proof script interactively, line-by-line, keeping track of what is to be proved, reporting fallacious steps and permitting

the user to correct errors and build proofs incrementally. Alternatively, Isabelle may check a collection of proof scripts non-interactively.

We have formalised a large proportion of the proof of the correspondence between OWL DL and OWL Full within Isabelle/HOL[15], and we are confident that the specified correspondence holds between the two semantics. We have also laid much of the groundwork for formalising further proofs of properties of OWL and RDF, such as the correspondence between the direct semantics and the RDF semantics of OWL DL ontologies.

We have not formalised the proofs of certain 'obvious' results, particularly those regarding the distinctness of BNodes, because it turns out that the usual line of informal reasoning hides a large amount of formal complexity and our time was better spent elsewhere. The 'impedance mismatch' between formal and informal arguments regarding distinctness of named variables is a general problem, not specifically anything to do with Isabelle/HOL, that current research elsewhere is trying to solve[16]. Instead of a formal proof in these cases, we appeal to the reader's understanding of the informal argument to justify the missing steps. The reader may use our published proof scripts to formally fill in the blanks, if desired: this task would be time-consuming, but should not be difficult.

The complexity of OWL and RDF is reflected in the complexity of the proof, which consists of over 60,000 lines of code. We will, obviously, not present the complete proof here, but will give a guide to the key points and methods that were used, with pointers to the relevant parts of the proof script for readers interested in seeing greater detail. We believe that it would be possible to translate our formal proof into an acceptably convincing paper proof simply by omitting some of the tinier details. The complete proof scripts are avilable at

`http://www.hpl.hp.com/techreports/2007/HPL-2007-146/sourcecode`

We used the version of Isabelle called `Isabelle2005: October 2005`, which is available for download from

`http://isabelle.in.tum.de/`

4 Definitions

4.1 Literals and Datatypes

A *plain literal* is either a string, or the ordered pair of a string and a language tag. Plain literals always denote themselves. A *typed literal* is the ordered pair of a string and a URI. A *literal* is either a plain literal or a typed literal.

A *datatype* d is a tuple $\langle v(d), lex(d), l_2v(d)\rangle$ where $v(d)$ is the set of values of d, $lex(d)$ is a set of strings representing the well-formed lexical representations of values of d, and $l_2v(d) : lex(d) \rightarrow v(d)$ takes lexical forms to their corresponding values.

Table 1. Abstract Syntax of data ranges, descriptions and individuals

```
  dataRange ::= DatatypeID(datatypeID)
              | rdfsLiteral
              | oneOf_d(dataLiteral*)

 cardinality ::= minCardinality(nat)
              | maxCardinality(nat)
              | cardinality(nat)

      dataRC ::= allValuesFrom_d(dataRange)
              | someValuesFrom_d(dataRange)
              | value_d(dataLiteral)
              | cardinality

individualRC ::= allValuesFrom_i(description)
              | someValuesFrom_i(description)
              | value_i(individualID)
              | cardinality

 restriction ::= restriction_d(datavaluedPropertyID
                   dataRC dataRC*)
              | restriction_i(individualvaluedPropertyID
                   individualRC individualRC*)

 description ::= ClassID(ClassID)
              | restriction
              | unionOf([description])
              | intersectionOf([description])
              | complementOf(description)
              | oneOf_i(individualID*)

  individual ::= individual(IndividualID? annotation*
                   type(description)* value*)

  annotation ::= annotation_uri(annotationPropertyID URI)
              | annotation_lit(annotationPropertyID dataLiteral)
              | annotation_ind(annotationPropertyID individual)

       value ::= value_uri(individualvaluedPropertyID individualID)
              | value_lit(datavaluedPropertyID dataLiteral)
              | value_ind(annotationPropertyID individual)
```

Table 2. Abstract syntax of facts, axioms and ontologies

```
fact ::= individual
       | SameIndividual(individualID individualID individualID*)
       | DifferentIndividuals(individualID individualID individualID*)

axiom ::= Class(classID Deprecated? (Complete | Partial)
            annotation* description*)
        | EnumeratedClass(classID Deprecated?
            annotation* individualID*)
        | DisjointClasses(description description description*)
        | EquivalentClasses(description description description*)
        | SubClassOf(description description)
        | Datatype(datatypeID Deprecated?
            annotation*)
        | DatatypeProperty(datavaluedPropertyID Deprecated?
            annotation* super(datavaluedPropertyID)* Functional?
            domain(description)* range(dataRange)*)
        | ObjectProperty(individualvaluedPropertyID Deprecated?
            annotation* super(individualvaluedPropertyID)*
            inverseOf(individualvaluedPropertyID)?
            Symmetric? Functional? InverseFunctional? Transitive?
            domain(description)* range(description)*)
        | AnnotationProperty(annotationPropertyID annotation*)
        | OntologyProperty(ontologyPropertyID annotation*)
        | EquivalentProperties_i(individualvaluedPropertyID
            individualvaluedPropertyID individualvaluedPropertyID*)
        | EquivalentProperties_d(datavaluedPropertyID
            datavaluedPropertyID datavaluedPropertyID*)
        | SubPropertyOf_i(individualvaluedPropertyID
            individualvaluedPropertyID)
        | SubPropertyOf_d(datavaluedPropertyID
            datavaluedPropertyID)

directive ::= axiom
            | fact
            | annotation_uri(annotationPropertyID URI)
            | annotation_lit(annotationPropertyID dataLiteral)
            | annotation_ind(annotationPropertyID individual)

            | annotation_ont(ontologyPropertyID ontologyID)

ontology ::= Ontology(ontologyID? directive*)
```

4.2 OWL Syntax and Semantics

The abstract-syntax representation of an OWL DL ontology is a collection of mutually recursive datatypes as defined in the OWL S&AS and summarised in tables 1 and 2. The theory `AbstractSyntax.thy` contains the translation of this definition into Isabelle. The shape of this definition is important, because much of the following work will be by induction over the abstract syntax, and the shape of a proof by induction is dictated by the shape of the associated recursive definition. In particular, notice that the datatypes `description`, `restriction` and `individualRC` are mutually recursive, as are the datatypes `annotation`, `value` and `individual`; therefore proofs regarding these datatypes will follow by mutual induction.

An *OWL vocabulary* is a tuple $\langle V_L, V_C, V_D, V_I, V_{DP}, V_{IP}, V_{AP}, V_{OP}, V_O, V_{AR} \rangle$ where V_L is a set of literals and the remaining components are sets of URIs comprising respectively the names of classes, datatypes, individuals, datatype properties, individual-valued properties, annotation properties, ontology properties, ontologies, and untyped referents of annotations. Some of the components have required members and some sets of components are mutually disjoint, as set out in the OWL S&AS.

The concept of a *direct OWL interpretation* is defined in the OWL S&AS. By slightly modifying this definition we obtain the conditions for a *infinite OWL interpretation*; both of these definitions are set out in table 3 and defined formally in `DirectSemantics.thy` and `NewDirectSemantics.thy` respectively. The difference of interest between direct OWL interpretations and infinite OWL interpretations is condition 17. The intention is that the semantics derived from infinite OWL interpretations corresponds more closely to the semantics of OWL Full (in which $O = EC(\texttt{owl:Thing}) = R$), and the differences between the two versions of condition 17 are 'almost' unobservable. Notice that the set O may be finite in a direct OWL interpretation, but it cannot be finite in an infinite OWL interpretation since $O = R \supseteq LV \supseteq \{l_p \mid l_p \text{ is a plain literal}\}$, and there are infinitely many plain literals. This motivates the name 'infinite', and highlights one of the known divergences between OWL Full and OWL DL.

There are other minor differences between direct and infinite OWL interpretations that arise from slight inconsistencies in the definitions of RDF and OWL which are unimportant in anything less than a totally formal proof. The two versions of condition 18 differ only on literals not in V_L, so it will not be possible to observe any difference between these conditions. Condition 19 is present to ensure it is possible to define an appropriate L in the presence of illformed literals. Notice that if there is an illformed literal in V_L then condition 18 implies condition 19, so this condition only has an effect on literals outside V_L. Again, this behaviour will not be observable.

Finally, we work in *HOL*, a strongly typed logic which has universal sets (of each type). For one of the constructions in the following work we will need to extend the universe R with a disjoint copy of LV, and this may not be possible if R is the universal set. Condition 20 ensures that this construction is

Table 3. Definition of OWL interpretations

A *direct OWL pre-interpretation* is a tuple $\langle V, D, R, EC, ER, L, S, LV, O\rangle$ where

- $V = \langle V_L, V_C, V_D, V_I, V_{DP}, V_{IP}, V_{AP}, V_{OP}, V_O, V_{AR}\rangle$ is an OWL vocabulary.
- D is a partial function from URIs to datatypes.
- R is a set of resources to be thought of as the 'universe' of the interpretation.
- $EC : \texttt{URI} \to \mathcal{P}R$ is a class-extension function.
- $ER : \texttt{URI} \to \mathcal{P}(R \times R)$ is a property-extension function.
- $L : \texttt{TypedLiteral} \to R$ interprets typed literals as resources.
- $S : \texttt{URI} \to R$ interprets URIs as resources.
- $LV \subseteq R$ is the set of all the literals in R.
- $O = EC(\texttt{owl:Thing}) \subseteq R$ is the set of all individuals in R.

and such that

1. $R \neq \varnothing$.
2. $\{l_p \mid l_p \text{ is a plain literal}\} \subseteq LV$.
3. For all d in the range of D, the value space $v(d) \subseteq LV$.
4. For all u in the domain of D, $u \in V_D$.
5. For all $u \in V_C$, $EC(u) \subseteq O$.
6. For all $u \in V_D$, $EC(u) \subseteq LV$.
7. For all $u \in V_{DP}$, $ER(u) \subseteq O \times LV$.
8. For all $u \in V_{IP}$, $ER(u) \subseteq O \times O$.
9. For all typed literals $l_t \in V_L$, $L(l_t) \in LV$.
10. For all $u \in V_I$, $S(u) \in O$.
11. $O \neq \varnothing$.
12. $EC(\texttt{owl:Nothing}) = \varnothing$.
13. $EC(\texttt{rdfs:Literal}) = LV$.
14. For all u in the domain of D, $EC(u) = v(D(u))$.
15. For all u in the domain of D and for all strings s, if $\mathit{Typed}(s, u) \in V_L$ then $L(\mathit{Typed}(s, u)) \in v(D(u))$.
16. For all u in the domain of D and for all strings s in the lexical space $\mathit{lex}(D(u))$, $L(\mathit{Typed}(s, u)) = l_2 v_{D(u)}(s)$.

A *direct OWL interpretation* is a direct OWL pre-interpretation such that additionally

17. $O \cap LV = \varnothing$
18. For all u in the domain of D and for all strings s not in the lexical space $\mathit{lex}(D(u))$, $L(\mathit{Typed}(s, u)) \in R \setminus LV$.

An *infinite OWL interpretation* is a direct OWL pre-interpretation such that additionally

17. $O = R$
18. For all u in the domain of D and for all strings s not in the lexical space $\mathit{lex}(D(u))$ such that $\mathit{Typed}(s, u) \in V_L$, $L(\mathit{Typed}(s, u)) \in R \setminus LV$.
19. $O \setminus LV \neq \varnothing$
20. There exists an injection with domain LV and range disjoint from R.

always possible. In a more traditional setting such as ZF, condition 20 is always satisfied.

5 Uncovering Errors

The act of formalising the prose of the OWL S&AS forced us to look at the specification extremely closely. In doing so, we discovered a number of minor errors in the document. Whilst none of these discoveries is especially important, and a reasonably knowledgable reader can see the intended meaning in each case, each bug can be seen to be non-trivial. This helps to validate our formal approach as a valuable one.

The most significant such discovery was that the mutual entailment between the two RDF graphs

$$\{\texttt{ex:foo} \quad \texttt{ex:bar} \quad \mathit{Typed}(\text{"}\texttt{10}\text{"}, \texttt{xsd:integer})\}$$

$$\{\texttt{ex:foo} \quad \texttt{ex:bar} \quad \mathit{Typed}(\text{"}\texttt{010}\text{"}, \texttt{xsd:integer})\}$$

if `xsd:integer` were a known datatype, does not hold, despite this being the content of the one of the RDF Test Cases[17]. The non-entailment arises if the vocabulary contains one, but not both, of the lexical forms of the number 10. We have published[18] and implemented the repair for this error, and also have received agreement from the W3C to publish our repair as an official erratum for the RDF Semantics[19].

We also discovered that the 'EC Extension Table' in section 3.2 of the OWL S&AS was missing a row describing the extension of the abstract syntax $\mathit{annotation}_{lit}(p\ l)$ where l is a literal. The same table also fails to cover certain degenerate cases, such as the extension of the empty intersection and of the syntax $\mathit{Individual}(\quad)$: both of these extensions should be O.

We note that in section 3.4 of the OWL S&AS the definition of satisfaction of an ontology is by recursion on the graph of `owl:imports`; however this relation is not forced to be well-founded so 'satisfaction' is ill-defined. The intended reading is to take the union of all the directives in the imports-closure of the ontology, which would permit `owl:imports` to contain loops or even to contain an infinite sequence.

Finally in section 5.2 of the OWL S&AS, in the subsection labelled 'Further conditions on `owl:oneOf`', we believe that if l is a sequence over LV_I and $\langle x, l\rangle \in EXT_I(S_I(\texttt{owl:oneOf}))$ then $x \in CEXT_I(\texttt{owl:DataRange})$, and also that $CEXT_I(\texttt{owl:DataRange}) \subseteq IDC$.

We have raised these three issues[20–22] as comments on OWL and believe that they will be addressed by the new OWL working group.

6 Statement of Lemmas and Theorems

The semantics of OWL ontologies is defined in terms of model theory in a conventional way. Write $\models_I a$ if an interpretation (or model) I satisfies an ontology

a and write $a \models b$ if a entails b, or in other words if for every I such that $\models_I a$ it follows that $\models_I b$. The various different semantics of OWL are obtained by varying the definitions of 'interpretation' and 'satisfies'.

The definition of satisfaction of an OWL ontology by an direct OWL interpretation is given in section 3.4 of the OWL S&AS. This definition rests on the definition of a direct OWL interpretation, but does not depend on the presence of condition 17 of that definition and hence can be appliee to to infinite OWL interpretations too. If a and b are OWL DL ontologies in abstract-syntax form then write $\models_I^{\mathrm{DL}} a$ and $a \models^{\mathrm{DL}} b$ for satisfaction and entailment with respect to direct OWL interpretations, and $\models_I^{\infty} a$ and $a \models^{\infty} b$ for satisfaction and entailment with respect to infinite OWL interpretations.

In order to convert an OWL DL ontology in abstract-syntax form into a corresponding RDF graph, it is necessary to generate a number of distinct blank nodes (BNodes) for use in the RDF graph. We perform this operation as a separate step, by decorating the abstract-syntax tree with distinct BNodes where appropriate. A decorated OWL DL ontology is assigned meaning essentially by removing the decorations and considering the semantics of the resulting undecorated ontology. Write $\models_I^{\infty+} a$ and $a \models^{\infty+} b$ for satisfaction and entailment of decorated ontologies a, b with respect to infinite OWL interpretations.

Finally, the RDF model theory[7] and the OWL S&AS together define the meaning of ontologies represented as RDF graphs. Write $\models^{\text{DL-RDF}}$ and $\models^{\text{Full}}$ for the DL and Full entailment relations respectively.

Let $\mathcal{A}$ be the set of all abstract-syntax representations of decorated OWL DL ontologies and $\mathcal{G}$ be the set of all RDF graphs. The conversion of a decorated ontology between its abstract-syntax representation and its RDF graph form is performed by a collection of nondeterministic mapping rules which define a relation $T \subseteq \mathcal{A} \times \mathcal{G}$, or equivalently a function $T : \mathcal{A} \to \mathcal{PG}$. Given this function, we may now state the semantic correspondence theorems quoted in the OWL S&AS.

Theorem 1 (OWL S&AS Theorem 1). *Subject to certain conditions on the vocabulary of interpretations, if A and B are imports-closed, then*

$$A \models^{DL} B \text{ if and only if } T(dec(A)) \models^{DL\text{-}RDF} T(dec(B))$$

Theorem 2 (OWL S&AS Theorem 2). *Subject to certain conditions on the vocabulary of interpretations, if A and B are imports-closed, then*

$$T(dec(A)) \models^{DL\text{-}RDF} T(dec(B)) \text{ implies that } T(dec(A)) \models^{Full} T(dec(B))$$

We aim to prove theorem 2; in the presence of theorem 1 it is equivalent to the following, which is the subject of our formal proof.

Theorem 3. *Subject to certain conditions on the vocabulary of interpretations, if A and B are imports-closed, then*

$$A \models^{DL} B \text{ implies that } T(dec(A)) \models^{Full} T(dec(B))$$

Our method of proof is to separate this implication into the following steps:

Lemma 1.

$$a \models^{DL} b \textit{ implies that } a \models^{\infty} b$$

Lemma 2.

$$a \models^{\infty} b \textit{ implies that } dec(a) \models^{\infty+} dec(b)$$

and that the BNodes of $dec(a)$ *and* $dec(b)$ *are all distinct.*

Lemma 3. *If the BNodes of* a *and* b *are all distinct then*

$$a \models^{\infty+} b \textit{ implies that } T(a) \models^{Full} T(b)$$

We have completely formalised lemma 1, showing that our 'infinite' semantics for OWL ontologies in abstract syntax form corresponds to the specified semantics. It is clear that lemma 2 holds, but our formalisation is only partial: working with BNodes is straightforward at an informal level but the arguments become very intricate when formalised, and for this reason we concentrated our efforts elsewhere.

We have also partially formalised lemma 3. The method of proof is the same as that followed in appendix A of the OWL S&AS, in particular in the proofs of lemmas 1, 1.1, 1.9, 2 and 3 there. A reader who is familiar with the lemmas of the OWL S&AS and the correspondence between the semantics of the abstract-syntax and the RDF representations of OWL DL ontologies should not be concerned with the omitted steps in this proof.

7 Outline of Proof

7.1 Lemma 1: Direct vs. Infinite semantics

We follow the conventional method to show a statement of the form of lemma 1; that is, we construct a function $\overline{(-)}$ from infinite OWL interpretations to direct OWL interpretations, such that $\models^{\infty}_{I} a$ if and only if $\models^{\mathrm{DL}}_{\overline{I}} a$. Then the conclusion follows, since if $\models^{\infty}_{I} a$ and $a \models^{\mathrm{DL}} b$ then $\models^{\mathrm{DL}}_{\overline{I}} a$ and hence $\models^{\mathrm{DL}}_{\overline{I}} b$ which implies that $\models^{\infty}_{I} b$ as required. So suppose that a is an OWL ontology in abstract-syntax form and let $I = \langle V, D, R, EC, ER, L, S, LV, O \rangle$ be a infinite OWL interpretation.

Recall that the main difference between infinite and direct OWL interpretations lies in condition 17 of their definitions. Suppose that the ontology a refers to a resource $r \in LV$: this could happen by explicitly mentioning a literal whose denotation is r, or by using some other syntax that just happens to denote r. For example, it may be that $S(\texttt{eg:x}) = r \in LV$, so that

$$\text{Individual}(\texttt{eg:x type}(\texttt{owl:Thing}))$$

denotes r. The syntax makes a reasonably strong distinction between this cases and the case where r is explicitly referred to as a literal, and we exploit this

distinction to define $\overline{I}$: we extend the universe R with a disjoint copy of LV and use the syntactic distinction to pick the appropriate copy of r in each case.

Let f be an injection whose range is LV and whose domain is disjoint from R, whose existence is ensured by condition 20 of the definition of an infinite OWL interpretation. It is this function f that captures what we mean by the action of 'copying' LV. Extend f to all of R by letting $f(r) = r$ for $r \in R \setminus LV$. Write $g`X = \{g(x) \mid x \in X\}$ and $(g \times h)(x, y) = (g(x), h(y))$. Define $\overline{I} = \langle V, D, \overline{R}, \overline{EC}, \overline{ER}, \overline{L}, \overline{S}, LV, \overline{O} \rangle$ as follows.

- $\overline{R} = R \cup f`LV$.
- If $u \in V_D$ then $\overline{EC}(u) = EC(u)$ else $\overline{EC}(u) = f`EC(u)$.
- If $u \in V_{DP}$ then $\overline{ER}(u) = (f \times 1)`ER(u)$.
- If $u \in V_{IP} \setminus V_{DP}$ then $\overline{ER}(u) = (f \times f)`ER(u)$.
- If $u \notin V_{IP} \cup V_{DP}$ then $\overline{ER}(u) = (f \times 1)`ER(u) \cup (f \times f)`ER(u)$.
- If l_t is a typed literal and $l_t \in V_L$ then $\overline{L}(l_t) = L(l_t)$.
- If $l_t = \mathit{Typed}(s, u)$ is a typed literal and $l_t \notin V_L$ and u is in the domain of D and $s \in \mathit{lex}(D(u))$ then $\overline{L}(l_t) = l_2 v_{D(u)}(s)$.
- If $l_t = \mathit{Typed}(s, u)$ is not covered by the preceding cases then $\overline{L}(l_t)$ is some arbitrary element of $\overline{R} \setminus LV$.
- $\overline{S}(u) = f(S(u))$.
- $\overline{O} = f`O$.

Then lemma `strengthen_strengthens` shows that $\overline{I}$ is a direct OWL interpretation, as required.

It remains to show that $\models^{\infty}_{I} a$ iff $\models^{\mathrm{DL}}_{\overline{I}} a$, subject to the following restrictions on the vocabulary V.

- V_C and V_D are disjoint,
- V_{DP}, V_{IP}, V_{AP} and V_{OP} are pairwise disjoint,
- `owl:Thing` $\in V_C$, and
- `rdf:type` $\notin V_{DP} \cup V_{IP}$.

The proof then proceeds by induction on the structure of a, using the following induction hypotheses:

- If d is a data range then $\overline{EC}(d) = EC(d)$.
- If r is a restriction then $\overline{EC}(r) = f`EC(r)$.
- If d is a description then $\overline{EC}(d) = f`EC(d)$.
- If ds is a list of descriptions then $\overline{EC}(\mathit{unionOf}(ds)) = f`EC(\mathit{unionOf}(ds))$.
- If ds is a list of descriptions then $\overline{EC}(\mathit{intersectionOf}(ds)) = f`EC(\mathit{intersectionOf}(ds))$.
- If a is an annotation then $\overline{EC}(a) \setminus LV = f`EC(a)$.
- If i is an individual construction then $\overline{EC}(i) = f`EC(i)$.
- If x is a fact then $\models^{\mathrm{DL}}_{\overline{I}} x$ iff $\models^{\infty}_{I} x$.
- If x is an axiom then $\models^{\mathrm{DL}}_{\overline{I}} x$ iff $\models^{\infty}_{I} x$.
- If a is an ontology annotation and $(o, S(\texttt{owl:Ontology})) \in ER(\texttt{rdf:type})$ then $\models^{\mathrm{DL}}_{\overline{I}} (f(o), a)$ iff $\models^{\infty}_{I} (o, a)$.

Finally, the lemma `Entails_implies_New_Entails` shows that if A and B are imports-closed sets of ontologies such that $A \models^{\mathrm{DL}} B$ then $A \models^{\infty} B$, as required.

7.2 Lemma 2: Infinite vs. Decorated Semantics

The decorator function $\mathrm{dec}(t, g, s)$ takes as input a syntax tree t, a 'gensym' function g and a list of already-generated BNodes s, and returns a decorated syntax tree t' and an updated state s'. We omit g, s and s' when they are unimportant, such as in the statement of lemma 2 above. We require that t and t' have the same semantics, and that the BNodes of t' are all distinct, in the sense defined in the theory `RDFTranslationTypes.thy`.

It is reasonably straightforward to see that t and t' have the same semantics, since the semantics of t' is defined to be that of the syntax obtained by removing the generated BNodes. In `DecoratedSemanticsMatch.thy` we have formalised the proof that if t and t' are data ranges, descriptions, individual constructions and annotations then $EC(t) = EC(t')$, but have not yet shown that the correspondence extends to facts and axioms. Given the relative ease of formalising the proof so far, we do not believe that this extension will present any difficulties.

It is also fairly straightforward to see that there exists a decorator which generates distinct BNodes. A partial proof of this fact is contained in the file `GenerateBNodesCorrect.thy` and runs as follows.

- Firstly, by induction on t we show that if $\mathrm{dec}(t, g, s) = (t', s')$ then $s \subseteq s'$.
- Secondly, also by induction on t we show that the BNodes of t' are contained within $s' \setminus s$.
- Finally, we show that the BNodes of t' really are distinct.

The first two steps have been fully formalised, and we have partially formalised the third step. Again, we do not believe that the remainder of the proof will present any difficulties.

7.3 Lemma 3: Decorated vs. RDF Semantics

We now wish to show that $a \models^{\infty+} b$ implies that $T(a) \models^{\mathrm{Full}} T(b)$. As in section 7.1, it suffices to show that if I is an OWL Full interpretation then there exists $\overline{I}$ such that $\models^{\mathrm{Full}}_{I} T(a)$ iff $\models^{\infty+}_{\overline{I}} a$.

Recall from the RDF model theory[7] that a simple RDF interpretation is a tuple $\langle P_I, R_I, V, EXT_I, S_I, L_I, LV_I \rangle$ where P_I, R_I and LV_I are sets of resources, the vocabulary V is a set of URIs and literals, the property extension function $EXT_I : P_I \to \mathcal{P}(R_I \times R_I)$, and S_I and L_I interpret URIs and literals respectively, subject to certain conditions. Define also $CEXT_I : R_I \to \mathcal{P}R_I$ by

$$CEXT_I(r) = \{x \mid (x, r) \in EXT_I(\texttt{rdf:type})\}.$$

From the OWL S&AS, an OWL Full interpretation is a simple RDF interpretation that satisfies a long list of extra conditions. It was straightforward to formalise these conditions, and the full details can be found in `RDFSemantics.thy`.

Say that an RDF interpretation I and a direct OWL pre-interpretation J are *corresponding* if the following conditions hold.

1. $LV = LV_I$.

2. $R = R_I$.
3. For all $u \in V_C \cup V_D$, $EC(u) = CEXT_I(S_I(u))$.
4. For all $u \in V_{DP} \cup V_{IP} \cup V_{AP} \cup V_{OP}$, $ER(u) = EXT_I(S_I(u))$.
5. For all typed $l_t \in V_L$, $L(l_t) = L_I(l_t)$.
6. For all r, $(r, S(\texttt{owl:DeprecatedClass})) \in ER(\texttt{rdf:type})$ iff $(r, S_I(\texttt{owl:DeprecatedClass})) \in EXT_I(\texttt{rdf:type})$.
7. For all r, $(r, S(\texttt{owl:DeprecatedProperty})) \in ER(\texttt{rdf:type})$ iff $(r, S_I(\texttt{owl:DeprecatedProperty})) \in EXT_I(\texttt{rdf:type})$.
8. For all $u \in V$, $S(u) = S_I(u)$.
9. The vocabulary V of J is *separated* in the sense that
   (a) V_C, V_D, V_I, V_{DP}, V_{IP}, V_{AP}, V_{OP}, V_O, and the disallowed vocabulary are pairwise disjoint.
   (b) V_C contains the built-in OWL classes.
   (c) V_D contains the datatypes of D and `rdfs:Literal`.
   (d) V_{AP} contains the built-in OWL annotation properties.
   (e) V_{OP} contains the built-in OWL ontology properties.
   (f) $V \setminus V_C$ contains none of the OWL class-only vocabulary.
   (g) $V \setminus V_D$ contains none of the OWL datatype-only vocabulary.
   (h) $V \setminus (V_{DP} \cup V_{IP} \cup V_{AP} \cup V_{OP})$ contains none of the OWL property-only vocabulary.
10. The vocabulary of I is the vocabulary of J together with the built-in OWL vocabulary.
11. The datatype map D of J contains `xsd:nonNegativeInteger`.
12. The vocabulary V of J is *satisfied* in the sense that
    - If $u \in V_C$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:Class}))$.
    - If $u \in V_D$ then $S_I(u) \in CEXT_I(S_I(\texttt{rdfs:Datatype}))$.
    - If $u \in V_I$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:Thing}))$.
    - If $u \in V_{DP}$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:DatatypeProperty}))$.
    - If $u \in V_{IP}$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:ObjectProperty}))$.
    - If $u \in V_{AP}$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:AnnotationProperty}))$.
    - If $u \in V_{OP}$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:OntologyProperty}))$.
    - If $u \in V_O$ then $S_I(u) \in CEXT_I(S_I(\texttt{owl:Ontology}))$.

Firstly, we show how to construct a infinite OWL interpretation $\overline{I}$ from an OWL Full interpretation $I = \langle P_I, R_I, V, EXT_I, S_I, L_I, LV_I \rangle$ and a datatype map D. We assume that V is separable, so that there exists an OWL vocabulary $V' = \langle V_L, V_C, \ldots \rangle$ such that $V' \cup V_{OWL} = V$, and that I satisfies V' in the sense defined above, where V_{OWL} is the built-in OWL vocabulary. We also assume the existence of an injection whose domain is LV_I and whose range is disjoint from R_I. In this case, letting

$$\overline{I} = \langle V', D, R_I, CEXT_I, EXT_I, L_I, S_I, LV_I, CEXT_I(S_I(\texttt{owl:Thing})) \rangle$$

gives a infinite OWL interpretation as required, as shown in the proof script `RDFIntToDirectInt.thy`.

It is now a rather trivial observation that I and $\overline{I}$ are corresponding interpretations in the sense defined above: see `RDFSemanticsCorrespond.thy` for details.

Finally, we show that if I and J are corresponding interpretations then $\models^{\text{Full}}_{I} T(a)$ iff $\models^{\infty+}_{J} a$. Because the mapping rule T is nondeterministic, $T(a)$ is a set of RDF graphs, and by $\models^{\text{Full}}_{I} T(a)$ we mean that for each $G \in T(a)$, $\models^{\text{Full}}_{I} G$.

The proof runs as two inductions over the structure of a, and follows the pattern of the proof given in appendix A of the OWL S&AS. Let $M(c)$ be the main node of the syntax fragment c, where defined. Firstly, we show that if $G \in T(a)$ then $\models^{\text{Full}}_{I} G$ implies that $\models^{\infty+}_{J} a$. Recall that if $\models^{\text{Full}}_{I} G$ then there exists a BNode-interpretation function A which assigns resoures to each BNode in G such that $\models^{\text{Full}}_{I+A} G$. We proceed by induction on the structure of a using the following induction hypotheses.

- If d is a data range then $(I+A)(M(d)) \in CEXT_I(S_I(\texttt{rdfs:Datatype}))$ and $CEXT_I((I+A)(d)) = EC(d)$.
- If d is a description then $(I+A)(M(d)) \in CEXT_I(S_I(\texttt{owl:Class}))$ and $CEXT_I((I+A)(d)) = EC(d)$.
- If i is an individual construct then $(I+A)(M(i)) \in EC(i)$.
- If annotation(p, x) is an annotation and $(y, (I+A)(M(x))) \in EXT_I(S_I(p))$ then $y \in EC(\text{annotation}(p, x))$.

We have formalised this far in the proof. It remains to show that if x is an axiom or fact and $\models^{\text{Full}}_{I+A} x$ then $\models^{\infty+}_{J} x$ which is essentially the content of Lemma 2 from Appendix A of the OWL S&AS. Thus $\models^{\text{Full}}_{I+A} G$ implies that $\models^{\infty+}_{J} a$, as required.

Conversely, we must show that if $\models^{\infty+}_{J} a$ then there exists a BNode interpretation function A such that $\models^{\text{Full}}_{I+A} G$. Firstly, we require that for any syntax fragment s, the BNodes of any $G \in T(s)$ are bounded by those of s, which is proved in `TranslationBNodePreservation.thy`, for data ranges, descriptions, individuals and annotations. Given this, we construct an A by making extensive use of the fact that the BNodes in each branch of the syntax tree of a are distinct. The proof runs by induction on the structure of a using the following induction hypotheses.

- If d is a data range and $G \in T(d)$ then there exists A such that $\models^{\text{Full}}_{I+A} G$ and such that A is defined for all BNodes in d.
- If d is a description and $G \in T(d)$ then there exists A such that $\models^{\text{Full}}_{I+A} G$ and such that A is defined for all BNodes in d.
- If i is an individual construction and $G \in T(i)$ and $r \in EC(i)$ then there exists A such that $\models^{\text{Full}}_{I+A} G$ and such that A is defined for all BNodes in d, and such that $(I+A)(M(i)) = r$.
- If annotation(p, x) is an annotation and $G \in T(\text{annotation}(p, x))$ and $r \in EC(\text{annotation}(p, x)))$ then there exists A such that $\models^{\text{Full}}_{I+A} G$ and such that A is defined for all BNodes in d, and such that

$$(r, (I+A)(M(x))) \in EXT_I(S_I(p)).$$

Again, this is as far as our formalisation effort has pushed. The remaining steps are again covered by Lemma 2 from Appendix A of the OWL S&AS. Thus, $a \models^{\infty+} b$ implies that $T(a) \models^{\text{Full}} T(b)$ as required.

8 Strengthening the Correspondence

So far in this discussion we have demonstrated that every entailment in the semantics of OWL DL is also an entailment of OWL Full. The converse of this result is false, as demonstrated by tests[11–14] in the OWL Test Case suite. However, these tests demonstrate the *only* known divergences between OWL DL and OWL Full, and it may be that a partial converse of the correspondence holds that captures exactly the collection of such divergences. Because of economic constraints we have made little progress towards a result of this nature, so we will conjecture a possible method and discuss its difficulties below. Firstly, on a positive note, we fully expect that the converse of lemma 2 holds. We now consider converses to lemmas 1 and 3.

8.1 A Converse to Lemma 1: Direct vs. Infinite semantics

The first pair of tests[11, 12] demonstrate that the class `owl:Thing` may be finite in OWL DL but not in OWL Full. We claim that it is possible to rewrite ontologies to permit `owl:Thing` to be infinite, by replacing all occurrences of `owl:Thing` with another name, say `my:Thing`, and for each class description that appears in the ontology, it is replaced by its intersection with EC(`my:Thing`). In short, if a user wishes to consider a finite universe, they are forced to work within a class other than the W3C-sanctioned `owl:Thing`. Notice that this is a purely syntactic transformation, that it is merely polynomial in the size of the ontology, and that it does not alter the expressive power of the language.

Additionally, one could insist that every ontology contains axioms that explicitly state that `owl:Thing` is infinite:

ObjectProperty(`ex:PredecessorOf` *InverseFunctional*
  range(`ex:PositiveInteger`))
individual(`ex:Zero` *type*(*complementOf*(`ex:PositiveInteger`))
  value(`ex:PredecessorOf`*Individual*( )))
SubClassOf(`ex:PositiveInteger`
  restriction(`ex:PredecessorOf` *someValuesFrom* `owl:Thing`))

In other words, `ex:Zero` is not a `ex:PositiveInteger`; `ex:Zero` has a successor, as does every `ex:PositiveInteger`, and no two have the same successor.

The second pair of tests[13, 14] show that, in OWL Full, if an object s is annotated by an object o then it is also annotated by an individual $o' \in$ `owl:Thing`, which follows from the trivial observation that $o \in$ `owl:Thing` for all o. Again, a polynomial syntactic transformation on the ontology forces this entailment to hold. We call this transformation a 'thingification'. Essentially, for every annotating literal, add a plain annotating anonymous individual; for every annotating URI u, add an annotating anonymous individual i with the same annotations as u.

Now we conjecture a partial converse to lemma 1, namely that if $a \models^{\infty} b$ then $a' \models^{\mathrm{DL}} b'$ where a' and b' are the respective results of transforming a and b by

the processes shown above. If this conjecture were true, this would demonstrate that the only observable semantic difference between $\models^{\infty}$ and $\models^{\mathrm{DL}}$ is that `owl:Thing` is infinite, and that annotation referents are all in `owl:Thing`. We have not attempted to prove this conjecture, as there are bigger obstacles in the way of demonstrating a relationship between $\models^{\mathrm{DL}}$ and $\models^{\mathrm{Full}}$, as discussed in the next section. We do not forsee any obstacles to a formal proof of this result, but have not even started to construct such a proof.

8.2 A Converse to Lemma 3: Decorated vs. RDF semantics

We would now like to show that if $T(a) \models^{\mathrm{Full}} T(b)$ then $a \models^{\infty+} b$, which would show that the model theory of infinite OWL interpretations gives rise to the semantics of OWL Full. To recap, the standard method is to take an infinite OWL interpretation I and construct an OWL Full interpretation J such that $\models_I^{\infty+} a$ iff $\models_J^{\mathrm{Full}} T(a)$. In fact, we know that if I and J are corresponding interpretations then the methods of section 7.3 show that $\models_I^{\infty+} a$ iff $\models_J^{\mathrm{Full}} T(a)$, so it is sufficient to take an infinite OWL interpretation I and attempt to construct a corresponding OWL Full interpretation J.

Unfortunately, it turns out that not every infinite OWL interpretation I has a corresponding OWL Full interpretation J. To see this, let p be a property and suppose that $\subseteq = S_J(\texttt{rdfs:subPropertyOf})$ and that $ER(p) = R \times R \setminus \{(\subseteq, p)\}$. If $(\subseteq, p) \in EXT_J(\subseteq)$ then, by the definition of `rdfs:subPropertyOf` it must be that $EXT_J(\subseteq) \subseteq EXT_J(S_J(p)) = ER(p)$, so that $(\subseteq, p) \notin EXT_J(\subseteq)$, which is a contradiction. If, however, $(\subseteq, p) \notin EXT_J(\subseteq)$ then, by the iff-condition for `rdfs:subPropertyOf`, it must be that $EXT_J(\subseteq) \not\subseteq EXT_J(S_J(p)) = ER(p)$ so that there exists $(x, y) \in EXT_J(\subseteq) \setminus ER(p)$. However, the only possible (x, y) is $(\subseteq, p)$, which is also a contradiction; thus there can be no property p with $ER(p) = R \times R \setminus \{(\subseteq, p)\}$. It is not hard to construct an interpretation I with enough properties like the p above to prevent any possible choice of $S_J(\texttt{rdfs:subPropertyOf})$, and such an I can have no corresponding OWL Full interpretation. It is likely that such a pathologically bad interpretation I could almost certainly be replaced by a more pleasant I' with the same entailments as I and which has a corresponding J, but it is not clear how one could construct such an I'. We have not demonstrated the result we seek is false, merely that a naïve application of the standard method fails. In some sense, it fails because of an inherent circularity in the semantics of OWL Full, namely that the iff-conditions for `rdfs:subPropertyOf` apply to all properties, including `rdfs:subPropertyOf` itself.

8.3 The Consistency of the RDF semantics

Given that it is difficult to construct an OWL Full interpretation that corresponds to a given infinite OWL interpretation, we turned to the apparently simpler problem of simply constructing any OWL Full interpretation I, which

would at least demonstrate that the RDF semantics of OWL Full is consistent. For now, we ignore the presence of literals. The comprehension conditions assert the existence of infinitely many elements of R_I, and by definition $R_I = CEXT_I(S_I(\texttt{owl:Thing})) = IOT$. We construct R_I by Herbrand's method: elements of the universe are syntax-trees x representing the collection of comprehension principles that assert the existence of x. For example,

```
[owl:Thing, restriction(owl:inverseOf allValuesFrom owl:Thing)]
```

is in IOT by the second comprehension principle for lists, since it is a list of elements of IOT (the first being obvious, and the second by the comprehension principle for `allValuesFrom` restrictions).

Having constructed R_I, we now seek to define the property-extension function EXT_I for each of the built-in properties, which includes `rdf:type`. Let $R_\in = EXT_I(S_I(\texttt{rdf:type}))$, and consider an attempt to define $R_\in$. Because class extensions are defined by recursion over the syntax of OWL, it would seem wise to define $R_\in$ by recursion too: in other words, define $CEXT_I(a) = \{b \mid (a,b) \in R_\in\}$ in terms of the class extensions of the components of the syntax-tree a. The class extension of

`restriction(rdf:type someValuesFrom` c`)`

is the union of the class extensions of the elements of $EC(c)$; thus to define this restriction we must know the extensions of each $x \in EC(c)$. But notice that we are proceeding by recursion over the syntax of c, not by $\in$-recursion, so we do not *a priori* know the extension of $x \in EC(c)$ and in particular either $EC(c)$ or its complement will contain c itself, so the obvious recursive techniques fail. In some sense, this failure is caused by the negative nature of taking complements which breaks the monotonicity of the recursive procedure, without which it is not immediately clear that recursion gives a well-defined result. From another viewpoint, permitting restrictions on `rdf:type` is a level of expressivity that may be pushing the bounds of the 'anyone can say anything about anything' mantra too far.

Now consider the construction of $R_{\text{cod}} = EXT_I(S_I(\texttt{rdfs:range}))$, and in particular note that the range of $R_\in$ is the collection of all nonempty classes. Thus R_{cod} cannot be defined before $R_\in$, because without $R_\in$ one does not know which are the nonempty classes. However, nor can $R_\in$ be defined first, because to do so would fix the class extension of restrictions such as

`restriction(rdfs:range hasValue` c`)`

before the class extension of c were known.

Similarly, consider the construction of $R_{\text{dom}} = EXT_I(S_I(\texttt{rdfs:domain}))$, and note that the domain of $R_\in$ is the collection of all classes whose extension is all of R_I; without knowning $R_\in$ one cannot define R_{dom}, but without knowing R_{dom} one cannot calculate the class extension of restrictions on `rdfs:domain`.

It gets worse: a restriction on `rdfs:domain` may or may not be nonempty, which affect `rdfs:range`. Similar circularities exist regarding `rdfs:subClassOf`

and `rdfs:subPropertyOf` too; thus all these built-in property extensions must be defined simultaneously, and this cannot happen by the obvious recursive technique. We do not assert that it is impossible to do so, but nor is it obviously possible and it was too intricate to achieve within the bounds of the time available to us.

8.4 Possible Repairs

It is hard to know exactly what to blame for the difficulty in this consistency proof. In one sense, it is the extensional 'if-and-only-if' semantics of the RDFS built-in properties that cause the circularities. If this were weakened to the intensional 'only if' style used by pure RDFS (and restrictions on `rdf:type` were suitably restricted) no feedback would occur and it would be more straightforward to use standard techniques to show the consistency of OWL. It would be possible to apply the intensional semantics just to the built-in properties themselves, and leave the extensional semantics in place elsewhere, which would have minimal impact on the OWL reasoners that are currently in use.

From another viewpoint, it is the comprehension principles themselves that are to blame. Carroll[23] suggests that taking a solipsistic stance and deleting the comprehension principles altogether would solve this problem. In this case, the only elements of the universe would be those that are explicitly mentioned.

It may, given sufficient time, be possible to complete the consistency proof for OWL as it stands, but it is likely that it would be hard to extend this proof to the correspondence result that we were originally seeking.

8.5 Consistency and OWL 1.1

At the time of writing, OWL 1.1 is under development and as yet has no RDF-compatible model theory. Two of its main extensions over OWL are as follows.

- Qualified cardinality restrictions (QCRs), which allow you to define a class such as
$$\{x \mid \mathrm{card}\{y \mid \langle x, y\rangle \in p \ \wedge \ y \in z\} \leq n\}$$
for properties p, classes z and integers n. Unqualified cardinality restrictions are similar, but do not have the condition that $y \in z$. An unqualified cardinality restriction on `rdf:type` is trivial, since every individual has infinitely many types, but QCRs on `rdf:type` are not as easy to deal with. Similarly, every property has infinitely many domains and ranges and every class has infinitely many (non-strict) superclasses, so unqualified cardinality restrictions on `rdfs:domain`, `rdfs:range` and `rdfs:subClassOf` are trivial, but QCRs on these properties are not.
- Subproperty chains, which allow you to make statements such as
$$p_1 \circ p_2 \circ \ldots \circ p_n \subseteq q$$

for properties $p_1, \ldots p_n$ and q. Note that `owl:sameAs` is the identity property, so that

$$p \circ ER(\texttt{owl:sameAs}) \subseteq p$$

for all properties p, and hence each property has infinitely many (non-strict) subproperties, which will complicate the definition of `rdfs:subPropertyOf`, especially in the presence of QCRs.

Because of these interactions, it is likely that any consistency proof for OWL 1.0 will be hard to directly extend to OWL 1.1.

9 Conclusions

We have designed a new semantics for OWL DL ontologies in abstract syntax form, and used this as a stepping-stone to showing the unproved correspondence between OWL DL entailment and OWL Full entailment. We have also developed machinery for further formal study of OWL within Isabelle/HOL. A paper proof of this correspondence could follow the method given here, but would almost certainly skip over certain 'obvious' details and may not have discovered the minor errors that we discussed in section 5. Although such discoveries cannot show that our formalisation within Isabelle/HOL is sound, it does provide some weight that our arguments are at least as valid as the paper proofs that missed these details. It is convenient that the definitions of RDF and OWL may be copied almost symbol-for-symbol into the language of Isabelle, which makes it straightforward to look at our formalised definitions and check that we have not 'cheated' at this stage. Given a reasonable confidence in the correctness of our definitions and in the veracity of Isabelle, we hope that the reader is convinced of our results.

We have also tried to demonstrate a partial converse to this result, to give a precise description of the difference between OWL DL and OWL Full. Unfortunately, this proof was too intricate to complete within our time constraints.

References[1]

1. Carroll, J.J., Klyne, G.: Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation, W3C (2004) `TR/2004/REC-rdf-concepts-20040210/`.
2. McGuinness, D.L., van Harmelen, F.: OWL web ontology language overview. W3C recommendation, W3C (2004) `TR/2004/REC-owl-features-20040210/`.
3. Lassila, O., Swick, R.R.: Resource description framework (RDF): Model and syntax specification. W3C recommendation, W3C (1999) `TR/1999/REC-rdf-syntax-19990222/`.

[1] For brevity, we have abbreviated the URIs that appear in the bibliography. Prepend `http://lists.w3.org/` to each that begins with the word `Archives`, and `http://www.w3.org/` to all the others.

4. Connolly, D., van Harmelen, F., Horrocks, I., McGuinnes, D.L., Patel-Schneider, P.F., Stein, L.A.: DAML+OIL (March 2001) reference description. W3C recommendation, W3C (2001) TR/2001/NOTE-daml+oil-reference-20011218.
5. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook. Cambridge University Press (2003)
6. Berners-Lee, T.: Web architecture: Metadata. Technical report, W3C (1997) DesignIssues/Metadata.html.
7. Hayes, P.: RDF semantics. W3C recommendation, W3C (2004) TR/2004/REC-rdf-mt-20040210/.
8. Patel-Schneider, P.F., Horrocks, I., Hayes, P.: OWL web ontology language semantics and abstract syntax. W3C recommendation, W3C (2004) TR/2004/REC-owl-semantics-20040210/.
9. Patel-Schneider, P.F., Horrocks, I.: OWL 1.1 web ontology language. W3C member submission, W3C (2006) Submission/2006/SUBM-owl11-overview-20061219/.
10. Smith, M.K.: Web ontology issue status. W3C issue status, W3C (2003) 2001/sw/WebOnt/webont-issues.html#I5.3-Semantic-Layering.
11. Carroll, J.: OWL test cases: Thing/004 (2004) TR/owl-test/byFunction#Thing-004.
12. Carroll, J.: OWL test cases: Thing/005 (2004) TR/owl-test/byFunction#Thing-005.
13. Carroll, J.: OWL test cases: AnnotationProperty/001 (2004) TR/owl-test/byFunction#AnnotationProperty-001.
14. Carroll, J.: OWL test cases: AnnotationProperty/002 (2004) TR/owl-test/byFunction#AnnotationProperty-002.
15. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL — A Proof Assistant for Higher-Order Logic. Volume 2283 of LNCS. Springer (2002)
16. Pitts, A.M.: Nominal logic, a first order theory of names and binding. Information and Computation **186** (2003) 165–193
17. Beckett, D., Grant, J.: RDF test cases. W3C recommendation, W3C (2004) TR/2004/REC-rdf-testcases-20040210/.
18. Carroll, J.J.: Comment on RDF model theory (2007) Archives/Public/www-rdf-comments/2007AprJun/0001.html.
19. Herman, I.: Errata for RDF semantics (2007) 2001/sw/RDFCore/errata.html#rdf-mt.
20. Turner, D.: Missing row from EC extension table (2007) Archives/Public/public-webont-comments/2007Apr/0001.html.
21. Turner, D.: Problem in definition of satisfaction, point 5 (2007) Archives/Public/public-webont-comments/2007Apr/0004.html.
22. Turner, D.: owl:oneOf and owl:DataRange (2007) Archives/Public/public-webont-comments/2007Aug/0003.html.
23. Carroll, J.: SEM solipsistic answers to Peter's entailments and paradox (2002) Archives/Public/www-webont-wg/2002Mar/0179.html.