On the Properties of Metamodeling in OWL

Boris Motik

FZI Research Center for Information Technologies at the University of Karlsruhe
Karlsruhe, Germany
motik@fzi.de

Abstract. A common practice in conceptual modeling is to separate the intensional from the extensional model. Although very intuitive, this approach is inadequate for many complex domains, where the borderline between the two models is not clear-cut. Therefore, OWL-Full, the most expressive of the Semantic Web ontology languages, allows combining the intensional and the extensional model by a feature we refer to as *metamodeling*. In this paper, we show that the semantics of metamodeling adopted in OWL-Full leads to undecidability of basic inference problems, due to free mixing of logical and metalogical symbols. Based on this result, we propose two alternative semantics for metamodeling: the *contextual* and the *HiLog* semantics. We show that $\mathcal{SHOIQ}$— a description logic underlying OWL-DL— extended with metamodeling under either semantics is decidable. Finally, we show how the latter semantics can be used in practice to axiomatize the logical interaction between concepts and metaconcepts.

1 Introduction

A common practice in conceptual modeling is to separate the *intensional* from the *extensional* model of a domain. The intensional model is analogous to a database schema and it describes the general structure and the regularities of the world. The extensional model is analogous to a database instance and it describes a particular state of the world. Such a modeling style has also influenced the design of the Ontology Web Language (OWL) [14], the W3C standard for building ontologies in the Semantic Web. Namely, OWL provides concepts and properties for building the intensional model, and individuals and relationships among them for building the extensional model.

To better understand this duality, consider the following example, originally presented in [16]; a similar example may be found in [15]. A natural way to represent kinship between animal species is to organize them in a hierarchy of concepts. For example, the concept *Bird* represents the set of all birds, and the concept *Eagle* is a subconcept of *Bird*, stating that all eagles are birds. This is an example of intensional knowledge, as it is concerned with defining the general notions of birds and eagles. Knowledge about concrete animals is represented by extensional knowledge, e.g. by stating that the individual *Harry* is an instance of *Eagle*. Now the intensional knowledge implies that *Harry* is an *Bird* as well.

However, one might also make statements about individual species, such as "eagles are listed in the IUCN Red List[1] of endangered species." Notice an important distinction: we do not say that each individual eagle is listed in the Red List, but that the eagle species as a whole is. Hence, we introduce a concept *RedListSpecies*, and consider the relationship between *RedListSpecies* and *Eagle*. Making the former a superconcept of the latter is incorrect, as it would imply that *Harry* is a *RedListSpecies* — clearly an undesirable conclusion. It is better to say that *Eagle* is a *type of RedListSpecies*. Thus, *RedListSpecies* acts as a *metaconcept* for *Eagle*. The style of modeling which provides for metaconcepts is called *metamodeling*, and it can be used to build concise models if we precisely axiomatize the properties of metaconcepts. For example, by stating that "it is not allowed to hunt the individuals of species listed in the Red List", we formalize the logical properties of the metaconcept *RedListSpecies*, allowing us to deduce that "it is not allowed to hunt *Harry*."

The examples such as the one given above are often dismissed with an argument that "eagle as a species" and "eagle as a set of all individual eagles" are not the one and the same thing, and should not be referred to using the same symbol. Whereas an in-depth philosophical investigation might provide a more definitive answer, we simply observe that the word "eagle" in most people's minds invokes a notion of a "mighty bird of prey." The interpretation of this notion as a concept or as an individual is secondary and is often context-dependent, so using different symbols for the same intuitive notion makes the model unnecessarily complex.

Metamodeling is provided in OWL-Full, the most expressive language of the OWL family. However, its semantics is controversial, mainly because it is non-standard, and therefore makes realizing practical reasoning systems difficult [5]. Therefore, OWL-DL was conceived as a "well-behaved" subset of OWL-Full by imposing the following restrictions: (*i*) the sets of logical and metalogical symbols are strictly separated, (*ii*) the sets of symbols used as concepts, roles and individuals are strictly separated, and (*iii*) restrictions required to yield a decidable logic, such as the one on simple roles in number restrictions [7], are enforced. These restrictions make OWL-DL a syntactical variant of the $\mathcal{SHOIN}(\mathbf{D})$ description logic, which is known to be decidable. This is desirable since, to practically implement reasoners for expressive logics, advanced optimization techniques are essential, and these are much easier to develop if the logic is decidable [1, ch. 9].

Since it does not enforce (*iii*), OWL-Full is trivially undecidable. To obtain a decidable logic supporting metamodeling, it is natural to ask whether OWL-DL, extended with metamodeling in the style of OWL-Full, remains decidable. However, in Section 2 we show that even the basic description logic $\mathcal{ALC}$ becomes undecidable if restrictions (*i*) and (*ii*) are not enforced.

We analyze this undecidability result, and show that it is actually due to (*i*), that is, to free mixing of logical and metalogical symbols. In a way, metamodeling in OWL-Full goes beyond its original purpose, and allows the user to tamper with

[1] http://www.redlist.org/

the semantics of the modeling primitives themselves. Therefore, in Section 3 we present two alternative semantics for metamodeling: a *contextual* or π-semantics, which is essentially first-order, and a *HiLog* or ν-semantics, which is based on HiLog [4] — a logic providing a second-order syntax for first-order logic. We show that, under some technical assumptions, both semantics can be combined with $\mathcal{SHOIQ}$, a description logic underlying OWL-DL, yielding a decidable fragment of OWL-Full without increasing the complexity of reasoning. Furthermore, we present a resolution-based decision procedure for the $\mathcal{SHIQ}$ fragment which, we believe, provides a basis for a practical implementation. Finally, in Section 4 we discuss the added expressivity of metamodeling on a concrete example. Technical details from this paper are presented in the technical report [9].

2 Undecidability of Metamodeling in OWL-Full

The semantics of OWL-Full [14] is quite technical, so we introduce $\mathcal{ALC}$-Full—an extension the basic description logic $\mathcal{ALC}$ with metamodeling in the style of OWL-Full. We use *rdf:*, *rdfs:* and *owl:* for the standard namespace prefixes.

Definition 1. *Let V be the* vocabulary *set consisting of these symbols:*

> *owl:Thing, owl:Nothing, rdf:type, rdfs:subClassOf, owl:sameAs, owl:differentFrom, owl:complementOf, owl:unionOf$_1$, owl:unionOf$_2$, owl:intersectionOf$_1$, owl:intersectionOf$_2$, owl:someValuesFrom, owl:allValuesFrom, owl:onProperty*

Let N be the set of names *such that $V \subseteq N$. An $\mathcal{ALC}$-Full* knowledge base *KB is a finite set of triples of the form $\langle s, p, o\rangle$, where $s, p, o \in N$.*

An interpretation *I is a triple $(\triangle^I, \cdot^I, EXT^I)$, where $\triangle^I$ is a non-empty set, $\cdot^I : N \to \triangle^I$ is a* name interpretation function *and $EXT^I : \triangle^I \to 2^{\triangle^I \times \triangle^I}$ is an* extension function*. Let $CEXT^I : \triangle^I \to 2^{\triangle^I}$ be the* concept extension function *defined as $CEXT^I(x) = \{y \mid (y, x) \in EXT^I(\textit{rdf:type}^I)\}$. An interpretation I is a* model *of KB if it satisfies all conditions from Table 1. KB is* satisfiable *if and only if a model of KB exists.*

$\mathcal{ALC}$-Full differs from OWL-Full in that: (*i*) it does not provide concrete predicates, (*ii*) it does not include the meta-level resources such as *owl:Class*, and (*iii*) it allows only binary union and intersection. These distinctions are not relevant for our undecidability proof. We use $\langle a \sqcup b, p, o\rangle$ as a syntactic shortcut for $\langle x, p, o\rangle$, $\langle x, \textit{owl:unionOf}_1, a\rangle$ and $\langle x, \textit{owl:unionOf}_2, b\rangle$, where x is a fresh name. We use similar shortcuts for $\langle s, p, a \sqcup b\rangle$ and for $\sqcap$.

We show the undecidability of $\mathcal{ALC}$-Full by a reduction from the well-known *domino tiling* problem [3]. A *domino system* is a triple $\mathcal{D} = (D, H, V)$, where $D = \{D_1, \ldots, D_n\}$ is a finite set of *domino types*, and $H \subseteq D \times D$ and $V \subseteq D \times D$ are *horizontal* and *vertical compatibility relations*, respectively. A *$\mathcal{D}$-tiling* of an infinite grid is a function $t : \mathbb{N} \times \mathbb{N} \to D$ such that $t(0, 0) = D_0$ and, for all $i, j \in \mathbb{N}$, $(t(i, j), t(i, j + 1)) \in H$ and $(t(i, j), t(i + 1, j)) \in V$. For an arbitrary domino system $\mathcal{D}$, determining whether a $\mathcal{D}$-tiling exists is undecidable [3].

Table 1. Semantics of $\mathcal{ALC}$-Full

1. $\langle s, p, o\rangle \in KB$ implies $(s^I, o^I) \in \mathrm{EXT}^I(p^I)$
2. $\mathrm{CEXT}^I(\mathit{owl{:}Thing}^I) = \triangle^I$
3. $\mathrm{CEXT}^I(\mathit{owl{:}Nothing}^I) = \emptyset$
4. $(x, y) \in \mathrm{EXT}^I(\mathit{rdfs{:}subClassOf}^I)$ implies $\mathrm{CEXT}^I(x) \subseteq \mathrm{CEXT}^I(y)$
5. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}sameAs}^I)$ implies $x = y$
6. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}differentFrom}^I)$ implies $x \neq y$
7. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}complementOf}^I)$ implies $\mathrm{CEXT}^I(x) = \triangle^I \setminus \mathrm{CEXT}^I(y)$
8. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}unionOf}_1^I)$ and $(x, z) \in \mathrm{EXT}^I(\mathit{owl{:}unionOf}_2^I)$ imply $\mathrm{CEXT}^I(x) = \mathrm{CEXT}^I(y) \cup \mathrm{CEXT}^I(z)$
9. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}intersectionOf}_1^I)$ and $(x, z) \in \mathrm{EXT}^I(\mathit{owl{:}intersectionOf}_2^I)$ imply $\mathrm{CEXT}^I(x) = \mathrm{CEXT}^I(y) \cap \mathrm{CEXT}^I(z)$
10. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}someValuesFrom}^I)$ and $(x, p) \in \mathrm{EXT}^I(\mathit{owl{:}onProperty}^I)$ imply $\mathrm{CEXT}^I(x) = \{w \mid (w, z) \in \mathrm{EXT}^I(p) \wedge z \in \mathrm{CEXT}^I(y)\}$
11. $(x, y) \in \mathrm{EXT}^I(\mathit{owl{:}allValuesFrom}^I)$ and $(x, p) \in \mathrm{EXT}^I(\mathit{owl{:}onProperty}^I)$ imply $\mathrm{CEXT}^I(x) = \{w \mid (w, z) \in \mathrm{EXT}^I(p) \rightarrow z \in \mathrm{CEXT}^I(y)\}$

For a domino system $\mathcal{D}$, let $KB_\mathcal{D}$ be the $\mathcal{ALC}$-Full knowledge base consisting of triples (1) – (9). Lemma 1 shows that satisfiability of $KB_\mathcal{D}$ exactly encodes the problem of deciding whether a $\mathcal{D}$-tiling exists.

$$\langle D_i \sqcap D_j, \mathit{rdfs{:}subClassOf}, \mathit{owl{:}Nothing}\rangle \text{ for } 1 \leq i < j \leq n \tag{1}$$

$$\langle \mathit{GRID}, \mathit{rdfs{:}subClassOf}, D_1 \sqcup \ldots \sqcup D_n\rangle \tag{2}$$

$$\langle \mathit{NotGRID}, \mathit{owl{:}complementOf}, \mathit{GRID}\rangle \tag{3}$$

$$\langle D_i, \mathit{rdfs{:}subClassOf}, \alpha_i\rangle, \langle \alpha_i, \mathit{owl{:}onProperty}, \mathit{owl{:}allValuesFrom}\rangle, \quad \langle \alpha_i, \mathit{owl{:}allValuesFrom}, \mathit{NotGRID} \sqcup \bigsqcup_{(D_i,d)\in H} d\rangle \text{ for } 1 \leq i \leq n \tag{4}$$

$$\langle D_i, \mathit{rdfs{:}subClassOf}, \beta_i\rangle, \langle \beta_i, \mathit{owl{:}onProperty}, \mathit{rdf{:}type}\rangle, \quad \langle \beta_i, \mathit{owl{:}allValuesFrom}, \mathit{NotGRID} \sqcup \bigsqcup_{(D_i,d)\in V} d\rangle \text{ for } 1 \leq i \leq n \tag{5}$$

$$\langle \mathit{GRID}, \mathit{owl{:}someValuesFrom}, \mathit{GRID}\rangle \tag{6}$$

$$\langle \mathit{GRID}, \mathit{owl{:}onProperty}, \mathit{owl{:}allValuesFrom}\rangle \tag{7}$$

$$\langle \mathit{GRID}, \mathit{owl{:}onProperty}, \mathit{rdf{:}type}\rangle \tag{8}$$

$$\langle \mathit{GRID}, \mathit{rdfs{:}subClassOf}, \mathit{owl{:}allValuesFrom}\rangle \tag{9}$$

$$\langle \mathit{rdf{:}type}, \mathit{owl{:}sameAs}, \mathit{owl{:}onProperty}\rangle \tag{10}$$

$$\langle a_{0,0}, \mathit{rdf{:}type}, \mathit{GRID} \sqcap D_0\rangle \tag{11}$$

Lemma 1. *A $\mathcal{D}$-tiling exists if and only if $KB_\mathcal{D}$ is satisfiable.*

Proof. ($\Rightarrow$) For a $\mathcal{D}$-tiling t, let I be an interpretation depicted in Figure 1, with $\mathrm{CEXT}^I(\mathit{GRID}^I) = \{a_{i,j}\}$ and $\mathrm{CEXT}^I(D_k^I) = \{a_{i,j} \mid t(i, j) = D_k\}$, for $i, j \geq 0$ and $1 \leq k \leq n$. The triples (3) – (5) encode the compatibility relations of $\mathcal{D}$ (including *NotGRID* into (3) and (4) ensures that compatibility is enforced only among instances of *GRID*). Hence, it is easy to see that I is a model of $KB_\mathcal{D}$.

($\Leftarrow$) Let I be a model of $KB_\mathcal{D}$. An excerpt of I is shown in Figure 1, in which a triple $\langle s, p, o\rangle$ is represented as an arc pointing from the node s to the

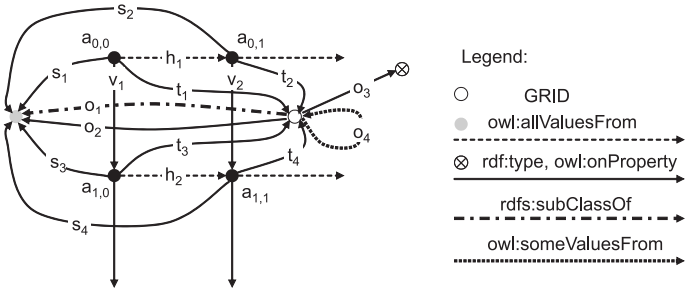

Fig. 1. Grid Structure in a Model of $KB_{\mathcal{D}}$

node o, whereas p is encoded by the line type according to the legend. To refer easily to arcs, we assign them labels t_i, h_i and v_i (these do not correspond to p). For example, the arc s_1 represents the triple $\langle a_{0,0}, \mathit{rdf{:}type}, \mathit{owl{:}allValuesFrom}\rangle$. Due to (10), *rdf:type* and *owl:onProperty* are synonyms, so s_1 also represents the triple $\langle a_{0,0}, \mathit{owl{:}onProperty}, \mathit{owl{:}allValuesFrom}\rangle$. By an abuse of notation, we do not distinguish between the symbols and their interpretations.

Due to (11), $a_{0,0}$ is linked by t_1 to *GRID*. Due to (6), (7) and (8), $a_{0,0}$ is linked to $a_{0,1}$ and $a_{1,0}$ through h_1 and v_1, respectively, and $a_{0,1}$ and $a_{1,0}$ are in the concept extension of *GRID* by t_2 and t_3, respectively. Due to (6) and (7), $a_{1,0}$ is linked by h_2 to $a_{1,1}$, and by t_4 to *GRID*. Finally, by (9), all $a_{i,j}$ are in the concept extension of *owl:allValuesFrom*, that is, all $a_{i,j}$ have an s_l arc to it.

Consider now the arcs at the node $a_{1,0}$. The arc s_3 can, due to (10), be read as $\langle a_{1,0}, \mathit{owl{:}onProperty}, \mathit{owl{:}allValuesFrom}\rangle$. By applying Item 11 of Table 1 for $x = a_{1,0}$ and $y = a_{1,1}$, we conclude that, if w is in the concept extension of $a_{1,0}$ and it is connected via $p = \mathit{owl{:}allValuesFrom}$ to some z, then z must be in the concept extension of $a_{1,1}$. However, we may now set $w = a_{0,0}$ due to v_1, and $z = a_{0,1}$ due to h_1; this implies that $a_{0,1}$ is in the concept extension of $a_{1,1}$, that is, that $a_{0,1}$ is connected to $a_{1,1}$ by v_2. Hence, $a_{0,0}$, $a_{0,1}$, $a_{1,0}$ and $a_{1,1}$ are arranged in a two-dimensional grid, which continues indefinitely due to (6) – (8).

A node $a_{i,j}$ in I is allowed to have multiple *owl:allValuesFrom* and *rdf:type* successors, and all $a_{i,j}$ need not be distinct, so I need not be a two-dimensional grid. However, a two-dimensional grid can easily be extracted from I: one can choose any *owl:allValuesFrom* successor $a_{i,j+1}$ and any *rdf:type* successor $a_{i+1,j}$ of $a_{i,j}$, as well as any *owl:allValuesFrom* successor $a_{i+1,j+1}$ of $a_{i+1,j}$. Regardless of the choices, $a_{i,j+1}$ is always connected to $a_{i+1,j+1}$ by *rdf:type*, so $a_{i,j}$, $a_{i,j+1}$, $a_{i+1,j}$ and $a_{i+1,j+1}$ are connected in a grid-like manner.

Hence, I contains a two-dimensional infinite grid in which *owl:allValuesFrom* are horizontal, and *rdf:type* are vertical arcs. The triples (1) – (5) ensure that each grid node is assigned a single domino type corresponding to the compatibility relations H and V of $\mathcal{D}$, so a $\mathcal{D}$-tiling can easily be constructed from I. □

Together with [3], Lemma 1 immediately implies the following result:

Theorem 1. *Checking satisfiability of an $\mathcal{ALC}$-Full knowledge base KB is undecidable.*

3 Two Decidable Approaches to Metamodeling

The proof of Lemma 1 reveals the causes for the undecidability of metamodeling in OWL-Full. Namely, this logic not only allows treating concepts as individuals, but it also allows mixing logical and metalogical symbols, and exposes its modeling primitives as individuals. We exploited this in axioms (5) and (6) of $KB_{\mathcal{D}}$, by stating an existential restriction on *owl:allValuesFrom* and *rdf:type* symbols and thus affecting their semantics. One would easily agree that tampering with the semantics of the ontology language is hardly desirable in practice, so in this section we present two alternative semantics for metamodeling.

In the following, we consider the description logic $\mathcal{SHOIQ}$, since it acts as the logical underpinning of the OWL family of languages. We do not consider datatypes here for the sake of simplicity. However, in [9] we show that, as long as datatypes are not subjected to metamodeling, they do not affect our results. We believe that this is not a practically relevant restriction: treating datatype individuals as concepts and vice versa will just unnecessarily confuse the users.

3.1 The Syntax and Semantics of $\mathcal{SHOIQ}$ with Metamodeling

Definition 2 (Syntax). *For N_a a set of* atomic names*, the set of* names *is defined as $N = N_a \cup \{n^- \mid n \in N\}$. For each $n \in N$, let $\mathsf{Inv}(n) = n^-$ and $\mathsf{Inv}(n^-) = n$. A $\mathcal{SHOIQ}$ RBox $KB_{\mathcal{R}}$ is a finite set of transitivity axioms $\mathsf{Trans}(R)$ and role inclusion axioms $R \sqsubseteq S$, where $R, S \in N$. As usual, we assume that $R \sqsubseteq S \in KB_{\mathcal{R}}$ implies $\mathsf{Inv}(R) \sqsubseteq \mathsf{Inv}(S) \in KB_{\mathcal{R}}$, and that $\mathsf{Trans}(R) \in KB_{\mathcal{R}}$ implies $\mathsf{Trans}(\mathsf{Inv}(R)) \in KB_{\mathcal{R}}$. Let $\sqsubseteq^*$ be the reflexive-transitive closure of $\sqsubseteq$. A name R is* simple *if for each name $S \sqsubseteq^* R$, $\mathsf{Trans}(S) \notin KB_{\mathcal{R}}$. A set of $\mathcal{SHOIQ}$* concepts *over $KB_{\mathcal{R}}$ is inductively defined as follows: each $A \in N$ is a concept and, for R and i names, S a simple name, C and D $\mathcal{SHOIQ}$ concepts and n a non-negative integer, $\{i\}$, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, $\forall R.C$, $\geq n\, R.C$ and $\leq n\, R.C$ are also $\mathcal{SHOIQ}$ concepts. A $\mathcal{SHOIQ}$ TBox $KB_{\mathcal{T}}$ is a finite set of* concept inclusion axioms *of the form $C \sqsubseteq D$, where C and D are $\mathcal{SHOIQ}$ concepts. A $\mathcal{SHOIQ}$ ABox $KB_{\mathcal{A}}$ is a finite set of* assertions *of the form $C(a)$, $R(a, b)$ or (in)equality axioms of the form $a \circ b$, where $\circ \in \{\approx, \not\approx\}$, C is a $\mathcal{SHOIQ}$ concept, and R, a and b are names. A $\mathcal{SHOIQ}$ knowledge base KB is a triple $(KB_{\mathcal{R}}, KB_{\mathcal{T}}, KB_{\mathcal{A}})$. The logic $\mathcal{ALCHOIQ}$ is a fragment of $\mathcal{SHOIQ}$ without the transitivity axioms. The logics $\mathcal{ALCHIQ}$ and $\mathcal{SHIQ}$ are the fragments of $\mathcal{ALCHOIQ}$ and $\mathcal{SHOIQ}$, respectively, without the nominal concepts $\{i\}$.*

The major difference of Definition 2 to the usual definitions is that the sets of concept, role and individual names are not disjoint, but are merged into one set of names. We denote with N_{KB} the subset of those names that occur in KB, and with $|KB|$ the *size* of KB with the numbers coded in unary. We now define the so-called *contextual* semantics for $\mathcal{SHOIQ}$.

Definition 3 (Contextual Semantics). *For a $\mathcal{SHOIQ}$ knowledge base KB, a π-interpretation I is a 4-tuple $(\triangle^I, \cdot^I, C^I, R^I)$ where $\triangle^I$ is a non-empty* domain

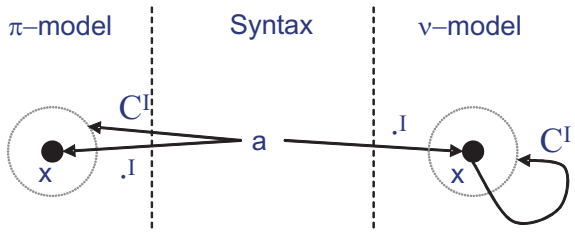

Fig. 2. π- and ν-models of the Axiom $a(a)$

set, $\cdot^I : N \to \triangle^I$ *is a* name interpretation function, $C^I : N \to 2^{\triangle^I}$ *is an* atomic concept extension function *and* $R^I : N \to 2^{\triangle^I \times \triangle^I}$ *is a* role extension function. *The function* C^I *is extended to concepts as specified in Table 2, upper left section, where symbols are interpreted* contextually*, that is, depending on their syntactic position. A* π*-interpretation* I *is a* π*-model of KB if it satisfies all conditions from Table 2, lower left section. The notions of* π*-satisfiability,* π*-unsatisfiability and* π*-entailment (written* $\models_\pi$*) are defined as usual.*

The contextual semantics is essentially equivalent to the one from [4] and to standard first-order semantics. Namely, in a first-order formula, the role of a symbol can be inferred from the place at which the symbol occurs in a formula, so the set of constant, function and predicate symbols need not be disjoint. We use π-semantics mainly as a baseline for a comparison with the HiLog semantics, defined below. This semantics is more in the spirit of OWL-Full, and is based on HiLog [4].

Definition 4 (HiLog Semantics). *For a* $\mathcal{SHOIQ}$ *knowledge base KB, a* ν*-interpretation* I *is a 4-tuple* $(\triangle^I, \cdot^I, C^I, R^I)$ *where* $\triangle^I$ *is a non-empty* domain set, $\cdot^I : N \to \triangle^I$ *is a* name interpretation function, $C^I : \triangle^I \to 2^{\triangle^I}$ *is an* atomic concept extension function*, and* $R^I : \triangle^I \to 2^{\triangle^I \times \triangle^I}$ *is a* role extension function. *The extension of the function* C^I *to concepts and the interpretation of axioms are specified in Table 2, right section. The notions of* ν*-satisfiability,* ν*-unsatisfiability and* ν*-entailment (written* $\models_\nu$*) are defined as usual.*

To understand the essential difference between these two semantics, consider the knowledge base KB containing only the axiom $a(a)$, where the symbol a is used both as an individual and as a concept. A π-model of KB is depicted on the left-hand side of Figure 2: both the individual interpretation $\cdot^I$ and the concept interpretation C^I are assigned directly to the symbol a. A ν-model of KB is depicted on the right-hand side of Figure 2: the individual interpretation $\cdot^I$ assigns the domain individual x to the symbol a; however, the concept interpretation is not assigned to a, but to x. We discuss the consequences that such a definition of semantics has on entailment in Section 4.

Neither semantics requires different names to be interpreted as different domain objects. If this is required, the *unique name assumption* should be axiomatized explicitly, by introducing an axiom $n_i \not\approx n_j$ for each $n_i, n_j \in N$, $n_i \neq n_j$.

Table 2. Two Semantics for $\mathcal{SHOIQ}$ with Metamodeling

π-semantics		ν-semantics
Extending C^I to concepts		C^I and the interpretation of axioms are obtained from the ones for π-semantics by applying the following changes: $C^I(A) \rightsquigarrow C^I(A^I)$ $R^I(S) \rightsquigarrow R^I(S^I)$ $R^I(T) \rightsquigarrow R^I(T^I)$ $R^I(\mathsf{Inv}(S)) \rightsquigarrow R^I(\mathsf{Inv}(S)^I)$
A	$C^I(A) \subseteq \triangle^I$	
$\{i\}$	$\{i^I\}$	
$\neg D$	$\triangle^I \setminus C^I(D)$	
$D_1 \sqcap D_2$	$C^I(D_1) \cap C^I(D_2)$	
$D_1 \sqcup D_2$	$C^I(D_1) \cup C^I(D_2)$	
$\exists S.D$	$\{x \mid (x,y) \in R^I(S) \wedge y \in C^I(D)$	
$\forall S.D$	$\{x \mid (x,y) \in R^I(S) \rightarrow y \in C^I(D)$	
$\leq n\, S.D$	$\{x \mid \sharp\{y \mid (x,y) \in R^I(S) \wedge y \in C^I(D)\} \leq n\}$	
$\geq n\, S.D$	$\{x \mid \sharp\{y \mid (x,y) \in R^I(S) \wedge y \in C^I(D)\} \geq n\}$	
Interpretation of axioms		
	$R^I(S) = R^I(\mathsf{Inv}(S))^-$	
$S \sqsubseteq T$	$R^I(S) \subseteq R^I(T)$	
$D_1 \sqsubseteq D_2$	$C^I(D_1) \subseteq C^I(D_2)$	
$\mathsf{Trans}(S)$	$R^I(S)^+ \subseteq R^I(S)$	
$D(a)$	$a^I \in C^I(D)$	
$S(a,b)$	$(a^I, b^I) \in R^I(S)$	
$a \approx b$	$a^I = b^I$	
$a \not\approx b$	$a^I \neq b^I$	

Note: $\sharp S$ is the number of elements in S, S^+ is the transitive closure of S, and S^- is the inverse relation of S.

Since the contextual semantics is essentially first-order, it can be decided using known algorithms, such as [8]. Therefore, we focus on deciding ν-satisfiability. In Subsection 3.2 we consider $\mathcal{ALCHOIQ}$ knowledge bases, in Subsection 3.3 we discuss the problems introduced by transitivity axioms, and in Subsection 3.4 we present a resolution-based practical decision procedure for the $\mathcal{ALCHIQ}$ fragment.

3.2 Deciding ν-satisfiability of $\mathcal{ALCHOIQ}$

An equivalence relation $\mathcal{E}$ over a set of names N induces a set or equivalence classes, so for each equivalence class we may arbitrarily select one *representative name* from it. For a name n, let $n/_{\mathcal{E}}$ denote the representative name chosen for the equivalence class of n, and for α an $\mathcal{ALCHOIQ}$ concept (axiom), let $\alpha/_{\mathcal{E}}$ denote the concept (axiom) obtained from α by replacing each name n with $n/_{\mathcal{E}}$. Finally, let KB be an $\mathcal{ALCHOIQ}$ knowledge base and $\mathcal{E}$ an equivalence relation over N_{KB}; then $KB/_{\mathcal{E}}$ is the knowledge base obtained from KB by (i) replacing each axiom α with $\alpha/_{\mathcal{E}}$, and by (ii) appending an axiom $n_i/_{\mathcal{E}} \not\approx n_j/_{\mathcal{E}}$ for each pair of names $n_i, n_j \in N_{KB}$ such that $n_i/_{\mathcal{E}} \neq n_j/_{\mathcal{E}}$.

An algorithm for checking ν-satisfiability of KB can be easily obtained by non-deterministically guessing an equivalence relation $\mathcal{E}$ over N_{KB}, and then by checking π-satisfiability of $KB/_{\mathcal{E}}$. The correctness of the algorithm is demonstrated by the following lemma:

Lemma 2. *An $\mathcal{ALCHOIQ}$ knowledge base KB is ν-satisfiable if and only if an equivalence relation $\mathcal{E}$ over N_{KB} exists, such that $KB/_{\mathcal{E}}$ is π-satisfiable.*

Proof. ($\Leftarrow$) Let $\mathcal{E}$ be an equivalence relation over N_{KB} and I^π a π-model of $KB/_{\mathcal{E}}$. We construct a ν-interpretation I_ν by setting $\triangle^{I_\nu} = \triangle^{I_\pi}$, $n^{I_\nu} = (n/_{\mathcal{E}})^{I_\pi}$, $C^{I_\nu}(n^{I_\nu}) = C^{I_\pi}(n/_{\mathcal{E}})$, $R^{I_\nu}(n^{I_\nu}) = R^{I_\pi}(n/_{\mathcal{E}})$, for each $n \in N_{KB}$ and, finally, $C^{I_\nu}(x) = C^{I_\pi}(x)$ and $R^{I_\nu}(x) = R^{I_\pi}(x)$ for each $x \in \triangle^{I_\nu}$ such that there is no $n \in N_{KB}$ with $n^{I_\nu} = x$. Due to inequality axioms $n_i/_{\mathcal{E}} \not\approx n_j/_{\mathcal{E}}$, we have $I_\pi(n_i/_{\mathcal{E}}) \neq I_\pi(n_j/_{\mathcal{E}})$, so the construction assigns a unique value to $C^{I_\nu}(x)$ and $R^{I_\nu}(x)$ for each $x \in \triangle^{I_\nu}$, and I_ν is correctly defined. Furthermore, for each concept X, $C^{I_\nu}(X) = C^{I_\pi}(X/_{\mathcal{E}})$, so I_ν is obviously a ν-model of KB.

($\Rightarrow$) Let I_ν be a ν-model of KB. We define $\mathcal{E} = \{(n_i, n_j) \mid {n_i}^{I_\nu} = {n_j}^{I_\nu}\}$ and construct a π-interpretation I_π by setting $\triangle^{I_\pi} = \triangle^{I_\nu}$, $(n/_{\mathcal{E}})^{I_\pi} = n^{I_\nu}$, $C^{I_\pi}(n/_{\mathcal{E}}) = C^{I_\nu}(n^{I_\nu})$ and $R^{I_\pi}(n/_{\mathcal{E}}) = R^{I_\nu}(n^{I_\nu})$, for each $n \in N_{KB}$. Again, for each concept X, $C^{I_\pi}(X/_{\mathcal{E}}) = C^{I_\nu}(X)$, so I_π is a π-model of $KB/_{\mathcal{E}}$. □

Now Lemma 2 immediately implies the following result:

Theorem 2. *Checking ν-satisfiability of an $\mathcal{ALCHOIQ}$ knowledge base KB can be performed in non-deterministic exponential time, assuming numbers are coded in unary.*

Proof. Observe that $|N_{KB}|$ is linear in $|KB|$, and that each equivalence relation $\mathcal{E}$ is a subset of $N_{KB} \times N_{KB}$. Hence, the number of possible equivalence relations is exponential in $|KB|$. A decision procedure for checking ν-satisfiability of KB can systematically examine all equivalence relations $\mathcal{E}$ and for each one perform a π-satisfiability check of $KB/_{\mathcal{E}}$. The last step can be performed in non-deterministic exponential time, since $\mathcal{ALCHOIQ}$ is a fragment of $\mathcal{C}^2$ — the two-variable first-order logic with counting, which is decidable in NExpTime assuming numbers are coded in unary [12]. Hence, the overall algorithm runs in non-deterministic exponential time as well. □

We briefly compare the results of Theorems 1 and 2. The main feature of ν-semantics is the reification of concept and role names. However, it is more like π-semantics and less like OWL-Full semantics in the way it handles the modeling primitives. In particular, in ν- and π-semantics, these are expressed as formulae and are not accessible as individuals in the knowledge base. On the contrary, OWL-Full reifies the modeling primitives as well, and thus allows their semantics to be altered by the statements in the knowledge base.

3.3 HiLog Semantics and Transitivity

The differences between the algorithms for checking ν- and π-satisfiability are minor. Since the latter algorithm can easily handle transitive roles, one might expect the former one to be easily extended to handle transitivity as well. However, consider the following knowledge base KB:

$$\top \sqsubseteq \geq 3\, S \tag{12}$$

$$S \approx T \tag{13}$$

$$\mathsf{Trans}(T) \tag{14}$$

Notice that KB is a $\mathcal{SHOIQ}$ knowledge base: the role S is simple, since it passes the syntactic criterion specified in Definition 2 (i.e., it is neither transitive nor it has transitive subroles). However, in any ν-interpretation I, (13) ensures that $S^I = T^I = \alpha$. Furthermore, due to (14), $R^I(\alpha)$ is transitive. Effectively, in (12) a transitive role is used in a number restriction, even though S is syntactically a simple role.

Since equality of role names might be non-trivially entailed by KB, identifying this requires theorem proving itself. This makes a check for simple roles under ν-semantics difficult, if not impossible. Because allowing transitive roles in number restrictions leads to undecidability [7], we get the following result:

Proposition 1. *Checking ν-satisfiability of a $\mathcal{SHOIQ}$ knowledge base KB is undecidable.*

Decidability can be regained by using *unique role assumption*, requiring symbols used as roles in KB to be interpreted as distinct domain individuals.

Definition 5 (Unique Role Assumption). *A $\mathcal{SHOIQ}$ knowledge base KB employs* unique role assumption *(URA) if it contains an axiom $S \not\approx T$ for each two distinct names S and T occurring as roles in KB.*

If KB employs URA or if it contains neither explicit equality statements nor number restrictions, role interpretations of different symbols can be assumed to be independent. Then, simple roles can be checked as usual, and transitivity axioms of KB can be eliminated by transforming KB into an equisatisfiable $\mathcal{ALCHOIQ}$ knowledge base $\Omega(KB)$, as done in [10]. Roughly speaking, a transitivity axiom $\mathsf{Trans}(S)$ is replaced with axioms of the form $\forall R.C \sqsubseteq \forall S.(\forall S.C)$, for each R with $S \sqsubseteq^* R$ and C a concept occurring in KB. This transformation is polynomial, so it does not increase the complexity of reasoning. Hence, ν-satisfiability of a $\mathcal{SHOIQ}$ knowledge base KB employing URA can be deiced by checking ν-satisfiability of the $\mathcal{ALCHOIQ}$ knowledge base $\Omega(KB)$.

3.4 A Practical Reasoning Procedure for $\mathcal{ALCHIQ}$

The reasoning procedure from Section 3.2 is worst-case optimal, but is unlikely to be effective in practice, since it systematically examines exponentially many equivalence relations. Therefore, we now present a practical, resolution-based algorithm for $\mathcal{ALCHIQ}$. It is an extension of our algorithm for deciding π-satisfiability from [10], and extending it to handle nomials is part of our ongoing work. Using the transformation of transitivity axioms from the previous subsection, this algorithm can also decide ν-satisfiability of $\mathcal{SHIQ}$ knowledge bases. Due to space constraints, we omit many technical details, which are given in [9]. We assume familiarity with first-order logic and resolution theorem proving.

Table 3. ν-semantics by Mapping into First-order Logic

Mapping Concepts to FOL		
$\nu_y(A, X)$	$=$	$\mathsf{isa}(A, X)$
$\nu_y(\neg D, X)$	$=$	$\neg\nu_y(D, X)$
$\nu_y(D_1 \sqcap D_2, X)$	$=$	$\nu_y(D_1, X) \wedge \nu_y(D_2, X)$
$\nu_y(D_1 \sqcup D_2, X)$	$=$	$\nu_y(D_1, X) \vee \nu_y(D_2, X)$
$\nu_y(\forall S.D, X)$	$=$	$\forall y : \mathsf{arole}(S, X, y) \rightarrow \nu_x(D, y)$
$\nu_y(\exists S.D, X)$	$=$	$\exists y : \mathsf{arole}(S, X, y) \wedge \nu_x(D, y)$
$\nu_y(\leq n\, S.D, X)$	$=$	$\forall y_1, \ldots, y_{n+1} : \bigwedge \mathsf{arole}(S, X, y_i) \wedge \bigwedge \nu_x(D, y_i) \rightarrow \bigvee y_i \approx y_j$
$\nu_y(\geq n\, S.D, X)$	$=$	$\exists y_1, \ldots, y_n : \bigwedge \mathsf{arole}(S, X, y_i) \wedge \bigwedge \nu_x(D, y_i) \wedge \bigwedge y_i \not\approx y_j$
Mapping Axioms to FOL		
$\nu(D_1 \sqsubseteq D_2)$	$=$	$\forall x : \nu_y(D_1, x) \rightarrow \nu_y(D_2, x)$
$\nu(S \sqsubseteq T)$	$=$	$\forall x, y : \mathsf{arole}(S, x, y) \rightarrow \mathsf{arole}(T, x, y)$
$\nu(D(a))$	$=$	$\nu_y(D, a)$
$\nu(S(a, b))$	$=$	$\mathsf{arole}(S, a, b)$
$\nu(a \circ b)$	$=$	$a \circ b$ for $\circ \in \{\approx, \not\approx\}$
Mapping KB to FOL		
$\nu(S)$	$=$	$\forall x, y : \mathsf{arole}(S, x, y) \leftrightarrow \mathsf{arole}(S^-, y, x)$
$\nu(KB)$	$=$	$\bigwedge_{\alpha \in KB_{\mathcal{R}} \cup KB_{\mathcal{T}} \cup KB_{\mathcal{A}}} \nu(\alpha) \wedge \bigwedge_{S \in N_{KB}} \nu(S)$

Notes:
(i): X is a meta variable and is substituted by the actual variable,
(ii): ν_x is defined as ν_y by substituting x, x_i and ν_x for y, y_i and ν_y, respectively.

Translation into First-order Logic. Our algorithm is based on resolution, so in Table 3 we define an operator ν which translates KB into a formula $\nu(KB)$ of first-order logic with equality. As shown by the following lemma, $\nu(KB)$ is first-order satisfiable if and only if KB is ν-satisfiable. Intuitively, for a name n, $\mathsf{isa}(n, x)$ encodes the concept extension of n and $\mathsf{arole}(n, x, y)$ encodes the role extension of n. Therefore, a ν-interpretation I_ν of KB can be easily converted into a first-order interpretation I of $\nu(KB)$ and vice versa.

Lemma 3. *For an $\mathcal{ALCHIQ}$ knowledge base KB, KB is ν-satisfiable if and only if a first-order model of $\nu(KB)$ exists.*

Basic Superposition Calculus. We decide first-order satisfiability of $\nu(KB)$ by *basic superposition* [2] ($\mathcal{BS}$), a clausal calculus optimized for theorem proving with equality. The calculus is parameterized by a certain term ordering and a selection function. It consists of resolution and superposition rules, which are applied only to literals in clauses designated by the chosen parameters. A set of clauses N is *saturated* by $\mathcal{BS}$ if applying a rule of $\mathcal{BS}$ to premises from N produces an already derived clause. $\mathcal{BS}$ is sound and complete: a saturated set of clauses N is unsatisfiable if and only if it contains the empty clause.

Decision Procedure by $\mathcal{BS}$. In order to apply $\mathcal{BS}$, we transform $\nu(KB)$ into the set of clauses $\Xi_\nu(KB)$ using the so-called *structural transformation* [11], which ensures that this step is polynomial.

We now saturate $\Xi_\nu(KB)$ by $\mathcal{BS}_{DL}$, where $\mathcal{BS}_{DL}$ denotes the $\mathcal{BS}$ calculus parameterized as discussed in [9]. It is possible to show that during such a saturation, only clauses of a certain syntactic form are derivable, and that the number of possible derived clauses is exponential in $|KB|$. Therefore, saturation by $\mathcal{BS}_{DL}$ will terminate after an exponential number of steps. Since $\mathcal{BS}_{DL}$ is sound and complete, it decides satisfiability of $\Xi_\nu(KB)$ and, by Lemma 3, ν-satisfiability of KB. The actual algorithm has to deal with several techical issues, for which we direct the reader to [9]. Hece we just state our main result:

Theorem 3. *For an $\mathcal{ALCHIQ}$ knowledge base KB, saturation of $\Xi_\nu(KB)$ by $\mathcal{BS}_{DL}$ decides ν-satisfiability of KB and runs in time exponential in $|KB|$, assuming numbers are coded in unary.*

4 Expressivity of Metamodeling

We now discuss the benefits of metamodeling in terms of additional consequences that can be drawn. These results are similar to the ones for HiLog from [4].

It is easy to see that ν-satisfiability is a strictly stronger notion than π-satisfiability. Consider the following knowledge base[2] KB:

$$Eagle(Harry) \tag{15}$$

$$\neg Aquila(Harry) \tag{16}$$

$$Eagle \approx Aquila \tag{17}$$

Under the contextual semantics, the interpretations of the symbols *Eagle* and *Aquila* as concepts and as individuals are independent, so KB is π-satisfiable. However, KB is ν-unsatisfiable: in each ν-interpretation $Eagle^I = Aquila^I = \alpha$, so it cannot be that $Harry^I \in C^I(Eagle^I)$ and $Harry^I \notin C^I(Aquila^I)$. For the other direction, we have the following lemma:

Lemma 4. *A ν-satisfiable $\mathcal{SHOIQ}$ knowledge base KB is also π-satisfiable.*

Proof. Let I_ν be a ν-model of an $\mathcal{SHOIQ}$ knowledge base KB. We construct a π-interpretation I_π as follows: $\triangle^{I_\pi} = \triangle^{I_\nu}$, $n^{I_\pi} = n^{I_\nu}$, $C^{I_\pi}(n) = C^{I_\nu}(n^{I_\nu})$ and $R^{I_\pi}(n) = R^{I_\nu}(n^{I_\nu})$, for each $n \in N_{KB}$. By a straightforward induction on the concept structure it can be shown that, for each concept X, $C^{I_\pi}(X) = C^{I_\nu}(X)$, so I_π is a π-model of KB. □

Furthermore, for a knowledge base with unique name assumption or without equality (either explicit or implicit, introduced through number restrictions), π-satisfiability and ν-satisfiability coincide:

Lemma 5. *Let KB be an $\mathcal{SHOIQ}$ knowledge base such that it employs unique name assumption, or it contains neither explicit equality statements nor number restrictions. Then KB is π-satisfiable if and only if it is ν-satisfiable.*

[2] "Aquila" is the Latin name for "eagle."

Proof. The ($\Leftarrow$) direction follows from Lemma 4. For the ($\Rightarrow$) direction, let KB be π-satisfiable in some model I_π. Since KB either employs unique name assumption or it does not employ equality, without loss of generality, we may assume that for each $n_i, n_j \in N$, $n_i \neq n_j$ implies ${n_i}^{I_\pi} \neq {n_j}^{I_\pi}$.

We now construct a ν-interpretation I_ν as follows: $\triangle^{I_\nu} = \triangle^{I_\pi}$, $n^{I_\nu} = n^{I_\pi}$, $C^{I_\nu}(n^{I_\nu}) = C^{I_\pi}(n)$ and $R^{I_\nu}(n^{I_\nu}) = R^{I_\pi}(n)$, for $n \in N_{KB}$. Furthermore, for all $x \in \triangle^{I_\nu}$ such that there is no $n \in N_{KB}$ with $x = n^{I_\nu}$, let $C^{I_\nu}(x) = R^{I_\nu}(x) = \emptyset$. Since we can assume that different names of N_{KB} are interpreted as different elements of $\triangle^{I_\nu}$, the construction assigns a unique value to $C^{I_\nu}(x)$ and $R^{I_\nu}(x)$ for each $x \in \triangle^{I_\nu}$, so I_ν is correctly defined. By a straightforward induction on the concept structure it can be shown that, for each concept X, $C^{I_\nu}(X) = C^{I_\pi}(X)$. Therefore, I_ν is a ν-model of KB. □

To summarize, ν-semantics allows deriving new consequences only if it is possible to derive that two symbols are equal; for example, from (15) and (17) it is possible to derive $Aquila(Harry)$. Furthermore, if the unique name assumption is employed, as it is often the case in practice, ν-semantics does not yield any additional consequences. This seems to suggest that the benefits of ν-semantics do not outweigh its drawbacks, namely, the fact that it is non-standard and that it introduces problems for transitive roles. Moreover, π-semantics might be sufficient for many practical applications.

However, ν-semantics unlocks its full potential when combined with a language more expressive than OWL. For example, by combining ν-semantics with the Semantic Web Rule Language (SWRL) [6], one can explicitly axiomatize the semantics of metaconcepts. Consider the example from Section 1. By (18) we state that *Eagle* is an *RedListSpecies*, and by a SWRL rule (19) we state that instances of species listed in the Red List are not allowed to be hunted. Notice that in atom $S(I)$ we use the variable S at the position of a predicate. Under ν-semantics this is equivalent to $\mathsf{isa}(S, I)$, but under π-semantics this would not be possible without leaving the confines of first-order logic. Now from (15), (18) and (19), we may infer $CannotHunt(Harry)$, so *RedListSpecies* semantically acts as a metaconcept of the *Eagle* concept.

$$RedListSpecies(Eagle) \tag{18}$$

$$RedListSpecies(S) \wedge S(I) \rightarrow CannotHunt(I) \tag{19}$$

To summarize, from the logical perspective, ν-semantics alone does not bring much, and π-semantics may be sufficient for numerous applications. However, ν-semantics provides a sound foundation for metamodeling, which, when combined with expressive logical formalisms such as SWRL, allows precisely axiomatizing the interaction between concepts and metaconcepts. Thus, we believe ν-semantics to be very relevant for the future extensions of OWL.

5 Related Work

The definition of ν-satisfiability given in Section 3 is inspired by HiLog [4], a logic in which general terms are allowed to occur in place of function and predicate

symbols in formulae. The semantics is defined by interpreting each individual as a member of the interpretation domain, and by assigning a functional and a relational interpretation to domain objects. The authors show that HiLog can be considered "syntactic sugar", since each HiLog formula can be encoded into an equisatisfiable first-order formula. The definition of the ν operator in Table 2 closely resembles this encoding. Finally, the authors show that a satisfiable first-order formula without equality is also satisfiable under HiLog semantics.

In [13], the RDFS Model Theory was criticized for allowing infinite number of meta-layers. The authors argue that such semantics is inadequate for the Semantic Web because (*i*) it does not provide adequate support for inferencing, (*ii*) it allows defining classes containing themselves, which may lead to paradoxes, and (*iii*) by adding classes, one necessarily introduces objects in the interpretation universe. The authors propose RDFS-FA, a stratified four-level approach, consisting of the meta-language layer, the language layer, the ontology layer and the instance layer. In [5] similar arguments were used to criticize the semantics of OWL-Full. We follow the principles of RDFS-FA by strictly separating the modeling primitives from the ontology and the instance layers. However, to allow metamodeling, our definition of ν-semantics merges the ontology and the instance layers into one. Furthermore, we show that (*iii*) affects the logical consequences only if equality reasoning is required, which matches well with the intuition behind metamodeling.

In [16] the authors point out the usefulness of metamodeling in many application domains. They propose separation of modeling layers, which are connected using so-called *spanning instances*. However, the authors do not consider the logical consequences of their approach.

6 Conclusion

In this paper we have analyzed the metamodeling features of OWL-Full, the most expressive of the Semantic Web ontology languages. We have shown that the style of metamodeling adopted in OWL-Full leads to undecidability of basic reasoning problems, due to mixing logical and metalogical primitives. In order to obtain a decidable and expressive language supporting metamodeling, we have proposed two alternative semantics: the contextual one, which is essentially first-order, and the HiLog one, which is more in the spirit of OWL-Full. Under certain technical assumptions, both semantics are decidable when combined with the description logic $\mathcal{SHOIQ}$. Furthermore, we have presented a practical resolution-based decision procedure for reasoning with $\mathcal{SHIQ}$ knowledge bases under HiLog semantics, thus obtaining practical support for a logic with metamodeling whose expressivity is between OWL-Lite and OWL-DL.

We have analyzed the added expressivity of metamodeling and have shown that the HiLog semantics allows deriving new conclusions only by equality reasoning. However, this approach unlocks its full potential if combined with expressive extensions, such as SWRL, since it allows axiomatizing the logical interaction between concepts and their metaconcepts.

In future, we shall attempt to extending the practical decision procedure from Subsection 3.4 to handle nominals as well, and thus to cover all of OWL-DL.

Acknowledgements

This work was partially funded by the EU IST project DIP 507483. We thank the anonymous reviewer for valuable comments regarding Subsection 3.2.

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, January 2003.
2. L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic Paramodulation. *Information and Computation*, 121(2):172–192, 1995.
3. R. Berger. The undecidability of the dominoe problem. *Memoirs of the American Mathematical Society*, 66, 1966.
4. W. Chen, M. Kifer, and D. S. Warren. HILOG: a foundation for higher-order logic programming. *Journal of Logic Programming*, 15(3):187–230, 1993.
5. I. Horrocks and P. F. Patel-Schneider. Three Theses of Representation in the Semantic Web. In *Proc. WWW 2003*, pages 39–47. ACM, 2003.
6. I. Horrocks and P. F. Patel-Schneider. A Proposal for an OWL Rules Language. In *Proc. WWW 2004*. ACM, 2004.
7. I. Horrocks, U. Sattler, and S. Tobies. Practical Reasoning for Very Expressive Description Logics. *Logic Journal of the IGPL*, 8(3):239–263, 2000.
8. I. Horrocks, U. Sattler, and S. Tobies. Reasoning with Individuals for the Description Logic $\mathcal{SHIQ}$. In *Proc. CADE 2000*, number 1831 in LNAI, pages 482–496. Springer, 2000.
9. U. Hustadt, B. Motik, and U. Sattler. Reasoning for Description Logics around $\mathcal{SHIQ}$ in a Resolution Framework. Technical Report 3-8-04/04, FZI, Karlsruhe, Germany, April 2004. http://www.fzi.de/ipe/publikationen.php?id=1172.
10. U. Hustadt, B. Motik, and U. Sattler. Reducing $\mathcal{SHIQ}^-$ Description Logic to Disjunctive Datalog Programs. In *Proc. KR 2004*, pages 152–162, Menlo Park, California, USA, June 2004. AAAI Press.
11. A. Nonnengart and C. Weidenbach. Computing Small Clause Normal Forms. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 6, pages 335–367. Elsevier Science, 2001.
12. L. Pacholski, W. Szwast, and L. Tendera. Complexity Results for First-Order Two-Variable Logic with Counting. *SIAM Journal on Computing*, 29(4):1083–1117, 2000.
13. J. Pan and I. Horrocks. RDFS(FA) and RDF MT: Two Semantics for RDFS. In *Proc. ISWC 2003*, number 2870 in LNCS, pages 30–46. Springer, 2003.
14. P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language; Semantics and Abstract Syntax. http://www.w3.org/TR/owl-semantics/, 2002.
15. G. Schreiber. The Web is not well-formed. *IEEE Intelligent Systems*, 17(2):79–80, March/April 2002. Contribution to the section "Trends & Controversies: Ontologies KISSES in Standardization", edited by S. Staab.
16. C. Welty and D. Ferrucci. What's in an instance? Technical Report 94-18, Max-Planck-Institut, 1994. RPI Computer Science.