

# ***KB\_Bio\_101* : A Challenge for OWL Reasoners**

Vinay K. Chaudhri, Michael A. Wessel, Stijn Heymans

SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94025-3493, USA  
firstname.lastname@sri.com

**Abstract.** We describe the axiomatic content of a biology knowledge base that poses both theoretical and empirical challenges for OWL reasoning. The knowledge base is organized hierarchically as a set of classes with necessary and sufficient properties. The relations have domain and range restrictions, are organized into a hierarchy, can have cardinality constraints and composition axioms stated for them. The necessary and sufficient properties of classes induce general graphs for which there are no known decidable reasoners. The OWL version of the knowledge base presented in this paper is an approximation of the original knowledge base. The knowledge content is practically motivated by an education application and has been extensively tested for quality.

## **1 Introduction**

The goal of Project Halo is to develop a “Digital Aristotle” - a reasoning system capable of answering novel questions and solving problems in a broad range of scientific disciplines and related human affairs [11]. As part of this effort, SRI has created a system called Automated User-Centered Reasoning and Acquisition System (AURA) [8], which enables educators to encode knowledge from science textbooks in a way that it can be used for answering questions by reasoning.

A team of biologists used AURA to encode a significant subset of a popular biology textbook that is used in advanced high school and introductory college courses in the United States [15]. The knowledge base called *KB\_Bio\_101* (for short: KBB101) is an outcome of this effort. KBB101 is a central component of an electronic textbook application called Inquire Biology [13] aimed at students studying from it.

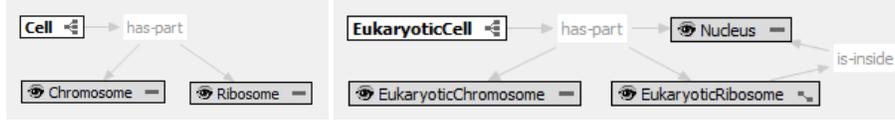
AURA uses a frame-based knowledge representation and reasoning system called Knowledge Machine (KM) [7]. We have translated the original KM-version of KBB101 into first-order logic with equality. By using this representation as a common basis, we have translated it into multiple different formats including SILK [9], OWL2 functional [17], answer set programming [5], and the TPTP FOF syntax [6]. In this paper, we describe the OWL2 translation. The translations are available for download [4].<sup>1</sup>

## **2 Modeling in the AURA Project – The Role of Skolem Functions**

AURA provides a graphical knowledge authoring environment for biologists. For example, the knowledge *Every Cell has a Ribosome part and a Chromosome part* is ex-

---

<sup>1</sup> This work is owned by Vulcan Inc. and is licensed for use under the *Creative Commons Attribution-NonCommercial-ShareAlike 3.0 license* (<http://creativecommons.org/licenses/by-nc-sa/3.0/>).



**Fig. 1.** (Simplified) Class Graphs for *Cell* and *EukaryoticCell* in AURA.

pressed graphically as shown in the left half of Fig. 1. Here, white nodes represent universally quantified variables, and grey nodes represent existentially quantified variables. *Cell* hence corresponds to the following first-order formula:

$$\forall x : Cell(x) \Rightarrow \exists y_1, y_2 : hasPart(x, y_1) \wedge hasPart(x, y_2) \wedge Ribosome(y_1) \wedge Chromosome(y_2)$$

Using the well-known technique of Skolemization, we can also write this as follows; the advantages of Skolem functions will become clear shortly:

$$\begin{aligned} \forall x : Cell(x) \Rightarrow \\ hasPart(x, f_{Cell}^1(x)) \wedge hasPart(x, f_{Cell}^2(x)) \wedge \\ Ribosome(f_{Cell}^1(x)) \wedge Chromosome(f_{Cell}^2(x)) \end{aligned}$$

The system supports inheritance. Consider the subclass *EukaryoticCell*, which inherits knowledge from *Cell*, see the right half of Fig. 1. The *Chromosome* in *EukaryoticCell* was inherited from *Cell*, and then specialized into a *EukaryoticChromosome*, and likewise for *Ribosome*. The *Nucleus* was added locally in *EukaryoticCell*. The advantage of using Skolem functions is that the inheritance can be made explicit by means of equality atoms: by adding  $f_{ECell}^3(x) = f_{Cell}^2(x)$ ,  $f_{ECell}^2(x) = f_{Cell}^1(x)$  to the Skolemized formula for *EukaryoticCell* it is made explicit that the *EukaryoticChromosome* in *EukaryoticCell* is a specialization of the *Chromosome* in *Cell* and consequently, all knowledge which was modeled for that *Chromosome* in the context of *Cell* also applies to the *EukaryoticChromosome* in the context of *EukaryoticCell* (in addition to what was modeled for *Chromosome* itself, of course):

$$\begin{aligned} \forall x : EukaryoticCell(x) \Rightarrow Cell(x) \wedge \\ hasPart(x, f_{ECell}^1(x)) \wedge hasPart(x, f_{ECell}^2(x)) \wedge hasPart(x, f_{ECell}^3(x)) \wedge \\ EukaryoticChromosome(f_{ECell}^3(x)) \wedge Nucleus(f_{ECell}^1(x)) \wedge \\ EukaryoticRibosome(f_{ECell}^2(x)) \wedge isInside(f_{ECell}^3(x), f_{ECell}^1(x)) \wedge \\ f_{ECell}^3(x) = f_{Cell}^2(x) \wedge f_{ECell}^2(x) = f_{Cell}^1(x) \end{aligned}$$

The employed graphical modeling paradigm can be described as *inherit*, *specialize*, and *extend*. During the modeling process, the system keeps track of the specialized and extended Skolem functions and records the inheritance structures as demonstrated.

Since the above axiom defines a graph, it is not expressible in the known decidable description logics [12].

### 3 The Axiomatic Content of *KB\_Bio\_101*

We first describe the signature of an AURA KB in first order logic. KBB101 is an AURA KB. Let  $CN$  be a set of class names (e.g.,  $Cell \in CN$ ), and  $RN$  be a set of relation names (e.g.,  $hasPart \in RN$ ). Let  $AN \subseteq RN$  be a set of attribute names (e.g.,  $color, temperature \in AN$ ). Let  $C, C_1, C_2, \dots, D, D_1, D_2, \dots, E, E_1, E_2, \dots, F, F_1, F_2, \dots$  be class names, and  $R, R_1, R_2, \dots, S, S_1, S_2, \dots, T, T_1, T_2, \dots$  be relation names. Let  $\{x, y, z, x_1, x_2, \dots\}$  be a set of variables, and, for every  $C \in CN$ , let  $\{fn_C^1, fn_C^2, \dots\}$

be a set of function symbols. We have the following sets of constants: *scalar constant values*  $SCs = \{small, big, \dots\}$ , *categorical constant values*  $CCs = \{blue, green, \dots\}$ , *cardinal unit classes*  $CUCs = \{meter, year, \dots\}$ , and  $CN \cup RN$  are considered constants as well. There are three kinds of attributes; they are used in so-called *value atoms*, see below:

**Cardinal attribute values:** For example, *t is 43 years* would be represented as  $age(t, t_1)$ ,  $theCardinalValue(t_1, 43)$ ,  $cardinalUnitClass(t_1, year)$ .

**Categorical attribute values:** For example, *t has color green* would be represented as  $color(t, t_1)$ ,  $theCategoricalValue(t_1, green)$ .

**Scalar attribute values:** For example, *t is big w.r.t. a house* (where house is a class) would be represented as  $size(t, t_1)$ ,  $theScalarValue(t_1, big)$ ,  $scalarUnitClass(t_1, house)$ .

Next we describe the axiomatic content. An *AURA KB* is a tuple  $(CTAs, CAs, RAs, EQAs)$ , where *CTAs* is a set of constant type assertions, *RAs* is a set of relation axioms, *CAs* is a set of class axioms, and *EQAs* is a set of equality atoms. Those axioms are described in the following:

**CTAs :** The AURA KB contains, for every  $c \in SCs \cup CCs \cup CUCs$ , 1 to  $n$  type assertions of the form  $C(c)$ , where  $C \in CN$  (the types of the constant).

**EQAs :** A set of equality atoms for  $C$ , of the form  $t = fn(t')$ , where  $t, t' \in \{x, fn_C^1(x), fn_C^2(x), \dots\}$ , and  $fn \in \{fn_D^1, fn_D^2, \dots\}$ , with  $C \neq D$ , for some  $D$  ( $D$  is a class mentioned in  $C$ , or a direct or indirect superclass of  $C$ ).

**CAs :** For every class name  $C \in CN$ , it may contain the following kinds of axioms: *DAs* : disjointness axioms:  $\forall x : C(x) \Rightarrow \neg D(x)$ ; *TAs* : taxonomic axioms:  $\forall x : C(x) \Rightarrow E(x)$ ; *NCAAs* : necessary conditions:  $\forall x : C(x) \Rightarrow \Phi[x]$ , where  $\Phi[x]$  is a conjunction of unary (class) atoms and binary (relation) atoms over terms  $\{x, fn_C^1(x), fn_C^2(x), \dots\}$ .

There are two special equality relations, namely *equal*, *notEqual*, which are user asserted equality atoms. The intended semantics is the semantics of first-order equality resp. in-equality. In order to distinguish them from the equalities in *EQAs* we use different predicate names.

Moreover,  $\Phi[x]$  can contain the following *value atoms*: for a term  $t$ , let *float* be a floating point number, *scalar*  $\in SCs$ , *categorical*  $\in CCs$ , *cardinalUnitClass*  $\in CUCs$ , and *scalarUnitClass*  $\in CN$ , then the following atoms are *value atoms*:  $theCardinalValue(t, float)$ ,  $theScalarValue(t, scalar)$ ,  $theCategoricalValue(t, categorical)$ ,  $cardinalUnitClass(t, cardinalUnitClass)$ , and  $scalarUnitClass(t, scalarUnitClass)$ .

In addition, an AURA KB can contain *qualified number restrictions*. Due to a lack of counting quantifiers, we represent them by means of quadrarity atoms  $maxCardinality(t, R, n, C)$  (resp.  $minCardinality$  and  $exactCardinality$ ), where  $n$  is a non-negative integer,  $C$  is a class, and  $R$  is a relation name.

**SCAs :** sufficient conditions:  $\forall x : \Theta[x, \dots] \Rightarrow C(x) \wedge EQs[x, \dots]$ , where  $\Theta[x, \dots]$  is a conjunction of unary, binary, value and qualified number restriction atoms over terms  $\{x, x_1, x_2, \dots\}$ , the sufficient conditions, and  $EQs[X, \dots]$  is a conjunction of equality atoms of the form  $t_1 = t_2$ , where  $t_1 \in \{x, x_1, x_2, \dots\}$ , and  $t_2 \in \{x, fn_C^1(x), fn_C^2(x), \dots\}$ , linking the variables in the antecedent to the Skolem function values in

the consequent of the necessary conditions,  $\Phi(x)$ . Obviously, requiring the use of the Skolem functions in the antecedent of the sufficient condition would be a too strong requirement and render the sufficient condition inapplicable in many cases. Also note that  $\Theta' [x] \subseteq \Phi [x]$ , where  $\Theta' [x]$  is the result of substituting the variables  $\Theta [x]$  with their respective Skolem terms from  $EQS [x, \dots]$ :  $\Theta' [x] = \Theta [x]_{\{t_1 \mapsto t_2, t_1 = t_2 \in EQS[x, \dots]\}}$ . Hence, every sufficient condition is also necessary (a byproduct of the graphical modeling).

For a given class name  $C$ , we refer to the corresponding axioms as  $DA_s(C)$ ,  $TA_s(C)$ , and  $EQA_s(C)$ . We refer to the union of all axioms for  $C$  as  $CA_s(C)$ .

**RA\_s** : For every relation name  $R \in RN$ ,  $RA_s$  may contain the following:  $DRA_s$  : relation domain restrictions  $\forall x, y : R(x, y) \Rightarrow C_1(x) \vee \dots \vee C_n(x)$ ;  $RRA_s$  : relation range restrictions  $\forall x, y : R(x, y) \Rightarrow D_1(y) \vee \dots \vee D_m(y)$ ;  $RHA_s$  : simple relation hierarchy  $\forall x, y : R(x, y) \Rightarrow S(x, y)$ ;  $QRHA_s$  : qualified relation hierarchy  $\forall x, y : R(x, y) \wedge C(x) \wedge D(y) \Rightarrow S(x, y)$ ;  $IRA_s$  : inverse relations  $\forall x, y : R(x, y) \Rightarrow S(y, x)$ ;  $12NA_s$  : 1-to-N cardinality  $\forall x, y, z : R(x, y) \wedge R(z, y) \Rightarrow x = z$ ;  $N21A_s$  : N-to-1 cardinality  $\forall x, y, z : R(x, y) \wedge R(x, z) \Rightarrow y = z$ ;  $TRANSA_s$  : simple transitive closure axioms  $\forall x, y, z : R(x, y) \wedge Rstar(y, z) \wedge C(x) \wedge D(y) \wedge E(z) \Rightarrow Rstar(x, z)$ , where  $Rstar(x, z) = R^*(x, z)$ ;  $GTRANSLA_s$  : generalized transitive closure axioms (left composition)  $\forall x, y, z : R(x, y) \wedge S(y, z) \wedge C(x) \wedge D(y) \wedge E(z) \Rightarrow Rstar(x, z)$ ; and  $GTRANSRA_s$  : generalized transitive closure axioms (right composition)  $\forall x, y, z : R(x, y) \wedge S(y, z) \wedge C(x) \wedge D(y) \wedge E(z) \Rightarrow Sstar(x, z)$ .

We refer to the axioms for a relation  $R$  by  $DRA_s(R)$  etc. We refer to the union of all axioms for  $R$  as  $RA_s(R)$ .

## 4 The OWL Translations of *KB\_Bio\_101*

Our OWL translator produces OWL2 functional syntax [17], which has good human readability and is readily processed by most OWL2 reasoners. The generated KBs have been syntax-tested with Protégé 4.2 [14] (utilizing the OWLAPI parser) as well as RacPro [16] (which has its own proprietary parser).

The following features of KBB101 might be challenging for OWL reasoners:

**Cycles**: KBB101 contains terminological cycles. It does not have the finite model property, nor the tree model property [1].

**Size**: the most complete export is 16 MBs big.

**Complexity**: the most complete export exploits  $SHOIQ(D_n)$  [3] (potentially we could use  $SROIQ(D_n)$  [10], but we currently do not include complex role inclusions, see below for a discussion).

**Graph structures**: we cannot represent the graph structures truthfully in OWL2. The original graph structures have to be approximated. We do this by rewriting and exporting the KBB101 in two flavors. **Flavor 1 - Unraveling**: We unravel the graph structures up to a certain maximal depth  $n$ . Unraveling is a standard technique from modal logics - let us give the following intuition: Given a class graph  $C$  (see Fig. 1), an up to max. depth  $n$  unraveled version of  $C$  can be produced by an  $n$ -bounded depth-first graph traversal of  $C$ , starting from the root node  $x$ , that outputs the (inverse) edge label whenever an (inverse) edge is traversed to visit a successor node, together with the node label. This produces a tree. This tree with max-depth  $n$  is then translated into OWL2 functional syntax as described. It results in an approximation of the original KBB101

which gets better the larger the value of  $n$  is. Note that nodes reachable only over paths of length  $> n$  are excluded. The filenames of KBB101s which were produced using unraveling start with `kb-owl-syntax-unraveled-depth-n`. We currently vary  $n$  from 0 to 4 and produce the corresponding KBB101s. With  $n = 0$ , the axioms in *NCA*s and *SCA*s are basically ignored, as the unraveled tree consists of the root node only (hence, only the taxonomy is exported). **Flavor 2 - Node IDs:** We can represent the graph structure by introducing symbolic *node identifiers* in the OWL2 class expressions. Even though the OWL2 reasoner will be blind to the intended semantic meaning of these node IDs, modeling graph structure and co-references, the original graph structure is at least represented and could, in principle, be exploited for reasoning by some powerful extended future OWL2 reasoner. Note that node IDs are only introduced if required (in tree-shaped class descriptions they are not required). Moreover, those node IDs can either be rendered as atomic classes, or introduced as nominals. The filenames of the respective KBB101s start with `kb-owl-syntax-coreference-IDs`.

**Explicit inheritance and equality:** the inter-class co-references between Skolem function values and equality atoms cannot be represented in OWL2. We hence skip all the axioms in *EQNs*. We consider the OWL2 export underspecified. In principle, we could preserve some of those by using functional properties and encoding tricks, but even then, feature agreements or role value maps might be required, and already *ALCF* with general TBoxes is undecidable [2].

**Rendering of axioms** We can en- and disable the export of certain axiom types, e.g. there is a switch which determines whether *DAs* are exported or not, and likewise for other axiom types. We produce all KBB101s for all possible combinations of those switches. Let us describe the rendering of class axioms and relation axioms. In the following,  $C'$  denotes the OWL2 version of class  $C$ , and  $R'$  the corresponding property of relation  $R$ .

The class axioms  $CA(C)$  are exported as follows: The axioms  $TAs(C)$  and  $NCA(C)$  are combined into one axiom of the form  $\forall x : C(x) \Rightarrow \Omega$ , which is then rendered as a `SubClassOf(C  $\Omega'$ )` axiom. Here,  $\Omega'$  is either an – up to depth  $n$  – unraveled version of  $\Omega$  as an OWL2 class expression, or the  $\Omega'$  class expression uses node IDs for representing the graph structure as described. Note that the  $DAs(C)$  and  $EQAs(C)$  are excluded here. Moreover, if  $C$  has a user-description or -comment, then this is rendered as `AnnotationAssertion(C' string)`. During rendering of *SCAs*, we are omitting the  $EQs[x, \dots]$  from  $\forall x : \Theta[x, \dots] \Rightarrow C(x) \wedge EQs[x, \dots] \in SCAs$ . KBB101s with *SCAs* preserved have a `triggers` in their file names. We generate a `SubClassOf( $\Theta'$  C)` axiom, where  $\Theta'$  is  $\Theta[x, \dots]$  as an OWL2 class expression, unraveled up to depth  $n$ . Disjointness axioms *DAs* are represented by means of `DisjointClasses`. The rendering of *DAs* can be suppressed; KBB101s with *DAs* preserved have `-disjointness` in their file names. The rendering of cardinality constraints in necessary conditions *NCA*s can be omitted. Also, we may choose to only export the cardinality constraints with cardinalities 0 and 1, as those are the only cardinality constraints used by the KM reasoning system [7] (the other cardinality constraints are ignored). KBB101s with cardinality constraints preserved have a `cardinalities` resp. `km-relevant-cardinalities` in their file names.

We also employ class and property annotation axioms to represent user descriptions and documentations.

The *inter-class equality axioms EQAs* are ignored – as explained, there is no straightforward way to model our Skolem function inheritance in OWL2. However, the user asserted *intra-class equality and in-equality atoms* are retained, and we are using the `:same-as` and `:not-equal` object properties for that purpose.

Exporting the relation axioms *RAs* is straightforward, too. KBB101s with relation axioms retained have a `relation-axioms` in their file names: The axioms *TRANSAs*, *GTRANSLAs*, *GTRANSRAs* are analyzed. If an axiom can be truthfully encoded as an OWL2 complex role inclusion axiom obeying the regularity condition [10], then it is included in the file (unfortunately, none are, so the KBB101 ends up in *SHOIQ(D<sub>n</sub>)* instead of *SR<sub>Q</sub>IQ(D<sub>n</sub>)*). If a relations *R* turns out to be transitive, then this is declared by means of `TransitiveObjectProperty(R)` axiom. *RDAs(R)* are rendered as `ObjectPropertyDomain(R, C)`, for every  $\forall x, y : R(x, y) \Rightarrow C(x) \in RDAs(R)$ . *RRAs(R)* are rendered as `ObjectPropertyRange(R, C)`, for every  $\forall x, y : R(x, y) \Rightarrow D(y) \in RRAs(R)$ . *RHAs(R)* are rendered as `SubObjectProperty(R, S)`, for every  $\forall x, y : R(x, y) \Rightarrow S(x, y) \in RHAs(R)$ . *IRAs(R)* are rendered as `InverseObjectProperties(R, S)`, for every  $\forall x, y : R(x, y) \Rightarrow S(y, x) \in IRAs(R)$ . If *N21As(R)*  $\neq \emptyset$ , then we declare `FunctionalObjectProperty(R)`, and `ObjectProperty(R)` otherwise. If *R* has a user-description string, then this is rendered as an `AnnotationAssertion(R string)`.

**Rendering of terms:** OWL2 is a term-free language. However, there is the analog of first-order constants, so-called nominals, and we may choose to use them for the representation of categorical property values (such as `green`) and scalar symbolic property values (such as `big`). A categorical property value such as `green` can either be represented as a type / instance assertion of the form `ClassAssertion(:ColorConstant :green)` and then used as a nominal object property filler in class sub-expressions such as `ObjectHasValue(:color :green)`, or `:green` might be a special subclass of `:ColorConstant`, `SubClassOf(:green :ColorConstant)`, and then used in an `ObjectSomeValuesFrom(:color :green)` expression to represent the color of some object. However, for string- and float-based property values we need to use a datatype property-based representation, e.g. `DataHasValue(:theCardinalValue "43.0e0"^^xsd:float)`. KBB101s using the nominal representation have a `value-nominals` in their file names, and otherwise `value-classes`. The rendering of value classes and nominals can also be switched off completely.

## 5 Conclusion

An initial version of the *KB\_Bio\_101* in OWL2 is now available [4] and we are very interested to actively engage with the research community to facilitate its use. We are also looking forward to seeing the reasoning runtimes of different systems participating in the ORE 2013 reasoner competition for the different OWL2 variants of KBB101. The reasoning problems we are currently interested in are consistency checking and classification.

**Acknowledgment:** This work has been funded by Vulcan Inc.

## References

1. F. Baader. Terminological Cycles in a Description Logic with Existential Restrictions. In *International Joint Conference on Artificial Intelligence*, 2003.
2. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
3. F. Baader and B. Hollunder. Qualifying Number Restrictions in Concept Languages. In *International Conference on Knowledge Representation and Reasoning*, 1991.
4. V. K. Chaudhri, S. Heymans, and M. A. Wessel. The *Bio\_KB\_101* Download Page, 2012. See <http://www.ai.sri.com/halo/halobook2010/exported-kb/biokb.html>.
5. V. K. Chaudhri, M. W. Stijn Heymans, and S. C. Tran. Object-Oriented Knowledge Bases in Logic Programming. Technical report, SRI International, 2013. Available from [http://www.ai.sri.com/pub\\_list/1935](http://www.ai.sri.com/pub_list/1935).
6. V. K. Chaudhri, M. A. Wessel, and S. Heymans. *KB\_Bio\_101*: A Challenge for TPTP First-Order Reasoners. In *CADE-24 Workshop on Knowledge Intensive Automated Reasoning*, 2013. Available from [http://www.ai.sri.com/pub\\_list/1937](http://www.ai.sri.com/pub_list/1937).
7. P. Clark and B. Porter. Building Concept Representations from Reusable Components. In *AAAI*. AAAI Press, 1997.
8. D. Gunning and V. Chaudhri et al. Project Halo Update Progress Toward Digital Aristotle. *AI Magazine*, Fall 2010.
9. B. Groszof. The SILK Project: Semantic Inferencing on Large Knowledge, 2012. See <http://silk.semwebcentral.org/>.
10. I. Horrocks, O. Kutz, and U. Sattler. The Even More Irresistible *SR<sub>O</sub>I<sub>Q</sub>*. In *International Conference on Knowledge Representation and Reasoning*, 2006.
11. V. Inc. Project Halo, 2012. See <http://www.projecthalo.com/>.
12. B. Motik, B. C. Grau, I. Horrocks, and U. Sattler. Representing Ontologies Using Description Logics, Description Graphs, and Rules. *Artificial Intelligence*, 173(14), Sept. 2009.
13. A. Overholtzer, A. Spaulding, V. K. Chaudhri, and D. Gunning. Inquire: An Intelligent Textbook. In *Proceedings of the Video Track of AAAI Conference on Artificial Intelligence*. AAAI Press, 2012. See [http://www.aaivideos.org/2012/inquire\\_intelligent\\_textbook/](http://www.aaivideos.org/2012/inquire_intelligent_textbook/).
14. Protégé Group. The Protégé Ontology Editor and Knowledge Acquisition System, 2012. See <http://protege.stanford.edu>.
15. J. B. Reece, L. A. Urry, M. L. Cain, S. A. Wasserman, P. V. Minorsky, and R. B. Jackson. *Campbell Biology, 9th ed.* Harlow: Pearson Education, 2011.
16. V. Haarslev and K. Hidde and R. Möller and M. Wessel. The RacerPro Knowledge Representation and Reasoning System. *Semantic Web Journal*, 3(3):267–277, 2012.
17. W3C OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009. Available at <http://www.w3.org/TR/owl2-overview/>.