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 (KB) is organized hierarchically as a set of classes with necessary and sufficient properties. The relations have domain and range constraints, are organized into a hierarchy, can have cardinality constraints and can have 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 KB presented in this paper is an approximation of the original KB. 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 [13]. As part of this effort, SRI has created a system called Automated User-Centered Reasoning and Acquisition System (AURA) [9], 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 [18]. The knowledge base called KB_Bio_101 (for short: KB) is an outcome of this effort. The KB is a central component of an electronic textbook application called Inquire Biology [1] aimed at students studying from it.

AURA uses a frame-based knowledge representation and reasoning system called Knowledge Machine (KM) [8]. We have translated the KM KB into first-order logic with equality. By using this representation as a common basis, we have translated the KB into multiple different formats including SILK [11], OWL2 description logics, answer set programming [6], and the TPTP FOF syntax [7]. We describe the OWL2 translation of the KB in this paper that is available for download [5].

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 Chromsome part* is expressed graphically as shown in the left half of the Fig. 1. Universally quantified node is shown

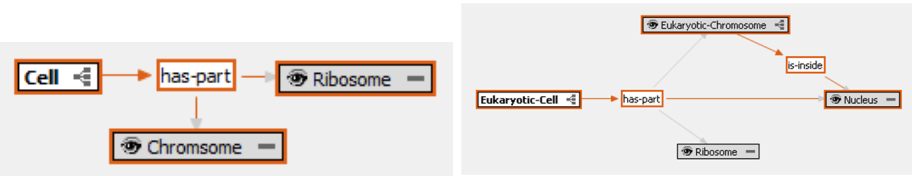

Fig. 1. Concept Graphs in AURA – Cell and EukaryoticCell.

in white and the existentially quanitifed nodes are shown in grey. This corresponds to the following first-order logic sentence:

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

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

$$\forall x : Cell(x) \Rightarrow$$
$$hasPart(x, f^1_{Cell}(x)) \wedge hasPart(x, f^2_{Cell}(x)) \wedge$$
$$Ribosome(f^1_{Cell}(x)) \wedge Chromsome(f^2_{Cell}(x))$$

The system supports inheritance. Consider the subclass $EukaryoticCell$, which inherits knowledge from $Cell$, see the right half of the Fig. 1. The $Chromosome$ in $EukaryoticCell$ was inherited from $Cell$, and then specialized into a $Eukaryotic\text{-}Chromosome$. Moreover, the $Ribosome$ was inherited from $Cell$ as well. 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: if we add $f^3_{ECell}(x) = f^2_{Cell}(x), f^2_{ECell}(x) = f^1_{Cell}(x)$ to the formula for $EukaryoticCell$. Doing so makes it clear that the $EukaryoticChromsome$ in $EukaroyticCell$ is a specialization of the $Chromsome$ in $Cell$ and consequently, every piece of knowledge which was modeled for that $Chromsome$ in the context of $Cell$ applies to the $EukaryoticChromsome$ in the context of $EukaryoticCell$ as well (in addition to what was modeled for $Chromsome$ itself, of course). Moreover, it is then clear that the $Ribosome$ is the same as the one inherited from $Cell$:

$$\forall x : EukaryoticCell(x) \Rightarrow$$
$$hasPart(x, f^1_{ECell}(x)) \wedge hasPart(x, f^2_{ECell}(x)) \wedge hasPart(x, f^3_{ECell}(x)) \wedge$$
$$EukaryoticChromsome(f^3_{ECell}(x)) \wedge Nucleus(f^1_{ECell}(x)) \wedge$$
$$Ribosome(f^2_{ECell}(x)) \wedge isInside(f^3_{ECell}(x), f^1_{ECell}(x)) \wedge$$
$$f^3_{ECell}(x) = f^2_{Cell}(x) \wedge f^2_{ECell}(x) = f^1_{Cell}(x)$$

Since the above axiom defines a graph, it is not expressible in the known decidable description logics [14]. 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.

3 The Axiomatic Content of Bio_KB_101

The original content of the AURA KB is best described on a first-order logic. We first describe the signature of the 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 $C, C_1, C_2, \ldots, D, D_1, D_2, \ldots, E, E_1, E_2, \ldots, F, F_1, F_2, \ldots$ be class names, and $R, R_1, R_2, \ldots, S, S_1, S_2, \ldots, T, T_1, T_2, \ldots$ be relation names. Let $\{x, y, z, x_1, x, \ldots\}$ be a set of variables, and, for every $C \in CN$, let $\{fn^1_C, fn^2_C, \ldots\}$ be a set of function symbols. We have the following sets of constants: *scalar constant values* $SCs = \{small, big, \ldots\}$, *categorical constant values* $CCs = \{blue, green, \ldots\}$, *cardinal unit classes* $CUCs = \{meter, year, \ldots\}$, 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 $age(t, t_1), theCardinalValue(t_1, 43), cardinalUnitClass(t_1, year)$.

Categorial 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 of the KB. 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 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 $t, t' \in \{x, fn^1_C(x), fn^2_C(x), \ldots\}$, and $fn \in \{fn^1_D, fn^2_D, \ldots\}$, with $C \neq D$, for some D (D is a class mentioned in C, or a direct or indirect superclass of C). Note that the maximum Skolem nesting depth is 2. Those describe the Skolem inheritance.

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: $\forall x : C(x) \Rightarrow E(x)$; $NCAs$: 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 $\{x, fn^1_C(x), fn^2_C(x), \ldots\}$.

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, the KB contains *qualified number restrictions.* Due to a lack of counting quantifiers, we represent them by means of quadrary atoms $maxCardinality(t, R, n, C)$, $minCardinality(t, R, n, C)$, and $exactCardinality(t, R, n, C)$, where n is a non-negative integer, C is a class, and R is a relation name.

SCAs : sufficient conditions: $\forall x : \Theta[x,\ldots] \Rightarrow C(x) \wedge EQs[x,\ldots]$, where $\Theta[x,\ldots]$ is a conjunction of unary, binary, value and qualified number restriction atoms over terms $\{x, x_1, x_2, \ldots\}$, the sufficient conditions, and $EQs[X,\ldots]$ is a conjunction of equality atoms of the form $t_1 = t_2$, where $t_1 \in \{x, x_1, x_2, \ldots\}$, and $t_2 \in \{x, fn^1_C(x), fn^2_C(x), \ldots\}$, 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 $\Theta'[x] \subseteq \Theta[x]$, where $\Theta'[x]$ is the result of substituting the variables $\Theta[x]$ with their respective Skolem terms from $EQs[x,\ldots]$: $\Theta'[x] = \Theta[x]_{\{t_1 \mapsto t_2, t_1 = t_2 \in EQs[x,\ldots]\}}$.Hence, every sufficient condition is also necessary.

For a given class name C, we refer to the corresponding axioms as $DAs(C)$, $TAs(C)$, and $EQAs(C)$. We refer to the union of all axioms for C as $CAs(C)$.

RAs : For every relation name $R \in RN$, RAs may contain the following: $DRAs$: relation domain restrictions $\forall x, y : R(x,y) \Rightarrow C_1(x) \vee \ldots \vee C_n(x)$; $RRAs$: relation range restrictions $\forall x, y : R(x,y) \Rightarrow D_1(y) \vee \ldots \vee D_m(y)$; $RHAs$: simple relation hierarchy $\forall x, y : R(x,y) \Rightarrow S(x,y)$; $QRHAs$: qualified relation hierarchy $\forall x, y : R(x,y) \wedge C(x) \wedge D(y) \Rightarrow S(x,y)$; $IRAs$: inverse relations $\forall x, y : R(x,y) \Rightarrow S(y,x)$; $12NAs$: 1-to-N cardinality $\forall x, y, z : R(x,y) \wedge R(z,y) \Rightarrow x = z$; $N21As$: N-to-1 cardinality $\forall x, y, z : R(x,y) \wedge R(x,z) \Rightarrow y = z$; $TRANSAs$: 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 $Rstar(x,z) = R^*(x,z)$; $GTRANSLAs$: 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 $GTRANSRAs$: 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 $DRAs(R)$ etc. We refer to the union of all axioms for R as $RAs(R)$.

4 The OWL Translations of Bio_KB_101

Our translator produces OWL2 KBs in functional syntax [15]. The OWL2 functional syntax has good human readability and is readily processed by most OWL2 reasoners. The generated KBs have been syntax-tested with Protégé 4.2 [16], Fact++ [10], as well as with RacerPro [17]. We are exporting the KB in different flavors, by including or omitting axioms of certain kinds as discussed see below.

The following features of the original KB make it challenging for OWL2 and OWL2 reasoners:

Cycles: the KB is cyclical, i.e., contains cyclical classes with refers-to cycles. It does not have the finite model property, nor the tree model property [2].

Size: the most complete export is 16 MBs big.

Complexity: the most complete export exploits $\mathcal{SHOIQ}(\mathcal{D})$ [4] (potentially we could use $\mathcal{SROIQ}(\mathcal{D}\backslash)$, [12] 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 ex-

porting the KB 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 which is not explained here in detail. It results in an approximation of the original KB which gets the better the larger the value of n is. The filenames of KBs which were produced using unraveling start with `kb-owl-syntax-unraveled-depth-n`. We are varying n from 0 to 4 and produce all those KBs. With $n = 0$, the axioms in $NCAs$ and $SCAs$ are ignored (basically. taxonomy only). **Flavor 2 - Node IDs:** We can represent the graph structure by introducing symbolic node identifiers as *node IDs* in the OWL2 class expressions. Even though the OWL2 reasoner will be blind to the intended semantic meaning of the node IDs, expressing 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 KBs 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 $\mathcal{ALCF}$ with general TBoxes is undecidable [3].

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 KBs for all possible combinations of those switches.

In the following, for a class C, `C'` denotes the OWL2 version, and likewise for relation R, `R'` denotes the corresponding property.

For every C, 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` Ω'`)` axiom. Ω' is either an – up to depth n – unraveled version of Ω as an OWL2 class, or the OWL2 class 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 an `AnnotationAssertion(C' string)`.

When rendering the $SCAs$, we are omitting the $EQs\,[x, \ldots]$ from $\forall x : \Theta\,[x, \ldots] \Rightarrow C(x) \wedge EQs\,[x, \ldots] \in SCAs$. KBs with $SCAs$ preserved have a `triggers` in their file names. We generate a `SubClassOf(`Θ' `C)` axiom, where Θ' is $\Theta\,[x, \ldots]$ 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. KBs with DAs preserved have a `-disjointness` in their file names.

The rendering of cardinality constraints in necessary conditions $NCAs$ can be omitted. Also, we may choose to only export the cardinality constraints with cardinalities 0 and 1; only those are relevant to the KM reasoning system which is the basis of AURA [8]. KBs with cardinality constraints preserved have a `cardinalities` resp. `km-relevant-cardinalities` in their file names.

Exporting the relation axioms RAs is mostly straightforward. KBs 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, then it is included in the file (unfortunately, none are, so the KB ends up in $\mathcal{SHOIQ}(\mathcal{D})$ instead of $\mathcal{SHROIQ}(\mathcal{D})$). 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 `Functional-ObjectProperty(R)`, and `ObjectProperty(R)` otherwise. If R has a user-description string, then this is rendered as an `AnnotationAssertion(R string)`.

We employ (class and property) annotation axioms to representing user description and documentation.

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.

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(:Color-Constant :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 `:Color-Constant, SubClassOf(:green :Color-Constant)`, 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(:the-cardinal-value "43.0e0"^^xsd:float)`. KBs 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 of completely.

5 Conclusion

An initial version of the KB_Bio_101 in OWL2 is now available [5] and we are very interested to actively engage with the research community to facilitate its use. Currently, all tested DL reasoners fail to check consistency of most of the non-trivial exports with $n > 1$ as soon as cardinality constraints are present, even only functional ones. It is actually unknown whether the bigger KBs with qualified number restrictions are consistent making it a good reasoning challenge.

Acknowledgment: This work has been funded by Vulcan Inc.

References

1. Adam Overholtzer and Aaron Spaulding and Vinay K. Chaudhri and Dave Gunning. Inquire: An Intelligent Textbook. In *Proceedings of the Vidoe Track of AAAI Conference on Artificial Intelligence*. AAAI Press, 2012. See `http://www.aaaivideos.org/2012/inquire_intelligent_textbook/`.
2. F. Baader. Terminological Cycles in a Description Logic with Existential Restrictions. In *International Joint Conference on Atificial Intelligence*, 2003.
3. 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.
4. F. Baader and B. Hollunder. Qualifying Number Restrictions in Concept Languages. In *International Conference on Knowledge Representation and Reasoning*, 1991.
5. V. K. Chaudhri, S. Heymans, and M. Wessel. The Bio_KB_101 Download Page, 2012. See `http://www.ai.sri.com/halo/halobook2010/exported-kb/biokb.html`.
6. 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`.
7. 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`.
8. P. Clark and B. Porter. Building Concept Representations from Reusable Components. In *AAAI*. AAAI Press, 1997.
9. D. Gunning and V. Chaudhri et al. Project Halo Update Progress Toward Digital Aristotle. *AI Magazine*, Fall 2010.
10. Dmitry Tsarkov and Ian Horrocks. Fact++, 2012. See `http://ow.man.ac.uk/factplusplus`.
11. B. Grosof. The SILK Project: Semantic Inferencing on Large Knowledge, 2012. See `http://silk.semwebcentral.org/`.
12. I. Horrocks, O. Kutz, and U. Sattler. The even more Irresistible $\mathcal{SROIQ}$. In *International Conference on Knowledge Representation and Reasoning*, 2006.
13. V. Inc. Project Halo, 2012. See `http://www.projecthalo.com/`.
14. 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.
15. B. Motik, P. F. Patel-Schneider, and B. Parsia. OWL2 Web Ontology Language, 2012. See `http://www.w3.org/TR/owl2-syntax/`.
16. Protégé Group. The Protégé Ontology Editor and Knowledge Acquisition System, 2012. See `http://protege.stanford.edu`.
17. Ralf Möller and Volker Haarslev and Kay Hidde and Michael Wessel. Racer Pro 2.0, 2012. See `http://www.racer-systems.com/products/racerpro/`.
18. J. B. Reece, L. A. Urry, M. L. Cain, S. A. Wasserman, P. V. Minorsky, , and R. B. Jackson. *Campbell Biology, 9th ed.* Benjamin Cummings, 2011.