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 Chromosome part is expressed graphically as shown in the left half of the Fig. 1. Universally quantified node is shown
in white and the existentially quantified nodes are shown in grey. This corresponds to
the following first-order logic sentence:
\[ \forall x : \text{Cell}(x) \quad \Rightarrow \quad \exists y_1, y_2 : \text{hasPart}(x, y_1) \land \text{hasPart}(x, y_2) \land \text{Ribosome}(y_1) \land \text{Chromosome}(y_2) \]

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:
\[ \forall x : \text{Cell}(x) \quad \Rightarrow \quad \text{hasPart}(x, f_1^\text{Cell}(x)) \land \text{hasPart}(x, f_2^\text{Cell}(x)) \land \text{Ribosome}(f_1^\text{Cell}(x)) \land \text{Chromosome}(f_2^\text{Cell}(x)) \]

The system supports inheritance. Consider the subclass \textit{EukaryoticCell}, which
inherits knowledge from \textit{Cell}, see the right half of the Fig. 1. The \textit{Chromosome} in
\textit{EukaryoticCell} was inherited from \textit{Cell}, and then specialized into a \textit{Eukaryotic-
Chromosome}. Moreover, the \textit{Ribosome} was inherited from \textit{Cell} as well. The 
Nucleus was added locally in \textit{EukaryoticCell}. The advantage of using Skolem functions is that
the inheritance can be made explicit by means of equality atoms: if we add
\[ f_3^\text{ECell}(x) = f_2^\text{Cell}(x) \land f_2^\text{ECell}(x) = f_1^\text{Cell}(x) \]
to the formula for \textit{EukaryoticCell}. Doing so makes it clear that the \textit{EukaryoticChromosome} in
\textit{EukaryoticCell} is a specialization of the \textit{Chromosome} in \textit{Cell} and consequently, every piece of knowledge which was modeled
for that \textit{Chromosome} in the context of \textit{Cell} applies to the \textit{EukaryoticChromosome} in
the context of \textit{EukaryoticCell} as well (in addition to what was modeled for \textit{Chromosome}

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 \textit{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., \textit{Cell} \( \in \) \( CN \)),
and \( RN \) be a set of relation names (e.g., \( \text{hasPart} \in RN \)). Let \( AN \subseteq RN \) be a set of
attribute names (e.g., color, temperature ∈ AN). Let C, C1, C2, ..., D, D1, D2, ..., E, E1, E2, ..., F, F1, F2, ... be class names, and R, R1, R2, ..., S, S1, S2, ..., T, T1, T2, ... be relation names. Let \{x, y, z, x1, x2, ...\} be a set of variables, and, for every C ∈ CN, let \{fn1_C, fn2_C, ...\} be a set of function symbols. We have the following sets of constants: scalar constant values SCs = \{small, big, ...,\}, categorical constant values CCs = \{blue, green, ...,\}, cardinal unit classes CUCs = \{meter, year, ...\}, and CN ∪ 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 

\(\text{age}(t, t_1), \text{theCardinalValue}(t_1, 43), \text{cardinalUnitClass}(t_1, \text{year})\).

**Categorical attribute values:** For example, t has color green would be represented as 

\(\text{color}(t, t_1), \text{theCategoricalValue}(t_1, \text{green})\).

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

\(\text{size}(t, t_1), \text{theScalarValue}(t_1, \text{big}), \text{scalarUnitClass}(t_1, \text{house})\).

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

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

\text{EQAs} : A set of equality atoms for \(C\), of the form \(t = fn(t')\), where \(t, t' \in \{x, fn1_C(x), fn2_C(x), ...\}\), and \(fn \in \{fn1_D, fn2_D, ...\}\), 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.

\text{CAs} : For every class name \(C \in \text{CN}\), it may contain the following kinds of axioms: 

\text{DAs} : disjointness axioms: \(\forall x : C(x) \Rightarrow \neg D(x)\); \text{TAs} : taxonomic axioms: \(\forall x : C(x) \Rightarrow D(x)\); \text{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 \(\{x, fn1_C(x), fn2_C(x), ...\}\).

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 \text{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 ∈ SCs, categorical ∈ CCs, cardinalUnitClass ∈ CUCs, and scalarUnitClass ∈ CN, then the following atoms are value atoms: 

\(\text{theCardinalValue}(t, \text{float}), \text{theScalarValue}(t, \text{scalar}), \text{theCategoricalValue}(t, \text{categorical}), \text{cardinalUnitClass}(t, \text{cardinalUnitClass}), \text{and scalarUnitClass}(t, \text{scalarUnitClass})\).

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: \( \forall x : \Theta [x,...] \Rightarrow C(x) \wedge EQs [x,...] \), where \( \Theta [x,...] \) is a conjunction of unary, binary, value and qualified number restriction atoms over terms \( \{x, x_1, x_2,...\} \), the sufficient conditions, and \( EQs [X,...] \) is a conjunction of equality atoms of the form \( t_1 = t_2 \), where \( t_1 \in \{x, x_1, x_2,...\} \), and \( t_2 \in \{x, fn^1(x), fn^2(x),...\} \), 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 \( EQs [x,...] : \Theta' [x] = \Theta [x]_{\{t_1\rightarrow t_2, t_1\rightarrow t_2 \in EQs[x,...]\}} \). 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: \( DAs: \) relation domain restrictions \( \forall x, y : R(x, y) \Rightarrow C_1(x) \lor ... \lor C_n(x); RAs: \) relation range restrictions \( \forall x, y : R(x, y) \Rightarrow D_1(y) \lor ... \lor D_m(y); RHAs: \) simple relation hierarchy \( \forall x, y : R(x, y) \Rightarrow S(x, y); QRHAs: \) qualified relation hierarchy \( \forall x, y : R(x, y) \land C(x) \land D(y) \Rightarrow S(x, y); IRAs: \) inverse relations \( \forall x, y : R(x, y) \Rightarrow S(y, x); \) \( 12NAs: \) 1-to-N cardinality \( \forall x, y, z : R(x, y) \land R(z, y) \Rightarrow x = z; N21As: \) N-to-1 cardinality \( \forall x, y, z : R(x, y) \land R(x, z) \Rightarrow y = z; TRANSAs: \) simple transitive closure axioms \( \forall x, y, z : R(x, y) \land Rstar(y, z) \land C(x) \land D(y) \land E(z) \Rightarrow Rstar(x, z) \), where \( Rstar(x, z) = R^*(x, z) \); \( GTRANSAs: \) generalized transitive closure axioms (left composition) \( \forall x, y, z : R(x, y) \land S(y, z) \land C(x) \land D(y) \land E(z) \Rightarrow Rstar(x, z); \) and \( GTRANSAs: \) generalized transitive closure axioms (right composition) \( \forall x, y, z : R(x, y) \land S(y, z) \land C(x) \land D(y) \land E(z) \Rightarrow Sstar(x, z) \).

We refer to the axioms for a relation \( R \) by \( DAs(R) \) 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 \( SHOIQ(D) \) [4] (potentially we could use \( SROIQ(D) \) \[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 ALC 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 $\text{SubClassOf}(C', \Omega')$ axiom. $\Omega'$ is either an – up to depth $n$ – unraveled version of $\Omega$ 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 $\text{AnnotationAssertion}(C' \text{ string}).$

When rendering the SCAs, we are omitting the $EQs[x,...]$ from $\forall x : \Theta[x,...] \Rightarrow C(x) \wedge EQs[x,...] \in SCAs$. KBs with SCAs preserved have a triggers in their file names. We generate a $\text{SubClassOf}(\Theta', C)$ axiom, where $\Theta'$ is $\Theta[x,...]$ as an OWL2 class expression, unraveled up to depth $n$.  

Disjointness axioms $DAs$ are represented by means of $\text{DisjointClasses}$. 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 $SHOIQ(D)$ instead of $SHROIQ(D)$). If a relations $R$ turns out to be transitive, then this is declared by means of $TransitiveObjectProperty\left(R\right)$ axiom. $RDAs\left(R\right)$ are rendered as $ObjectPropertyDomain\left(R, C\right)$, for every $\forall x,y: R\left(x,y\right) \Rightarrow C\left(x\right) \in RDAs\left(R\right)$. $RRAs\left(R\right)$ are rendered as $ObjectPropertyRange\left(R, C\right)$, for every $\forall x,y: R\left(x,y\right) \Rightarrow D\left(y\right) \in RRAs\left(R\right)$. $RHAs\left(R\right)$ are rendered as $SubObjectProperty\left(R,S\right)$, for every $\forall x,y: R\left(x,y\right) \Rightarrow S\left(x,y\right) \in RHAs\left(R\right)$. $IRAs\left(R\right)$ are rendered as $InverseObjectProperties\left(R, S\right)$, for every $\forall x,y: R\left(x,y\right) \Rightarrow S\left(y,x\right) \in IRAs\left(R\right)$. If $N21As\left(R\right) \neq \emptyset$, then we declare $Functional-ObjectProperty\left(R\right)$, and $ObjectProperty\left(R\right)$ otherwise. If $R$ has a user-description string, then this is rendered as an $AnnotationAssertion\left(R \text{ } \text{ string}\right)$.

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 $\text{same-as}$ and $\text{not-equal}$ 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 $\text{green}$) and scalar symbolic property values (such as $\text{big}$). A categorical property value such as $\text{green}$ can either be represented as a type / instance assertion of the form $\text{ClassAssertion}\left(:\text{Color-Constant } \text{green}\right)$ and then used as a nominal object property filler in class sub-expressions such as $\text{ObjectHasValue}\left(:\text{color } \text{green}\right)$, or $\text{green}$ might be a special subclass of $:\text{Color-Constant}$, $\text{SubClassOf}\left(:\text{green } :\text{Color-Constant}\right)$, and then used in an $\text{ObjectSomeValuesFrom}\left(:\text{color } :\text{green}\right)$ 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. $\text{DataHasValue}\left(:\text{the-cardinal-value } \text{"43.0e0"} \text{"xsd:float}\right)$. 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