KB_Bio_101: A Challenge for TPTP First-Order Reasoners

Vinay K. Chaudhri, Michael A. Wessel, Stijn Heymans.
Artificial Intelligence Center, SRI International, Menlo Park, CA, 94025

Abstract

We describe the axiomatic content of a biology knowledge base that poses both theoretical and empirical challenges for knowledge intensive reasoning. The knowledge base is organized hierarchically as a set of classes with necessary and sufficient properties. The class hierarchy also contains disjointness axioms. 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 knowledge content is practically motivated by an education application and has been extensively tested to be of high quality.

1. Introduction

The goal of Project Halo is to develop a ``Digital Aristotle'' - a reasoning system capable of answering novel questions and solving advanced problems in a broad range of scientific disciplines and related human affairs [1]. As part of this effort, SRI has created a system called Automated User-Centered Reasoning and Acquisition System (AURA) [2], 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 [3]. 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* [4] aimed at students studying from it. SRI has worked with teachers and students to collect a large number of questions that are of practical interest for this application. Working from those questions, the team has formulated reasoning tasks that must be performed by a reasoner.

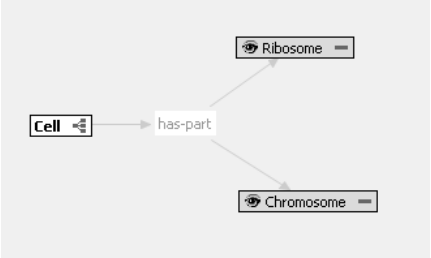
AURA uses a frame-based knowledge representation and reasoning system called Knowledge Machine (KM) [5]. We are able to translate the KM KB into first-order logic with equality [6]. By using this representation as a common basis, we can translate the KB into multiple different formats including SILK [7], OWL2 description logics [8], answer set programming [9], and the TPTP FOF syntax [10,11]. We describe the TPTP FOF translation of the KB in this paper.

The *KB_Bio_101* presents a unique opportunity for empirical and theoretical research on Knowledge intensive reasoning methods.

The paper is structured as follows. We first describe how graph-structured classes are modeled in the AURA project to give a flavor of the knowledge base, and we describe how Skolem functions and equality atoms are used to capture the inheritance structure. Next we describe the systematic knowledge engineering processes by which the KB was produced. We then enumerate the axiom schemas that appear in it. Finally, we discuss the TPTP export and some stats of KB in this syntax. We conclude with an informal description of interesting reasoning tasks for this KB to challenge the community.

2. Modeling in the AURA Project – The Role of Skolem Functions and Equalities

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 follows:



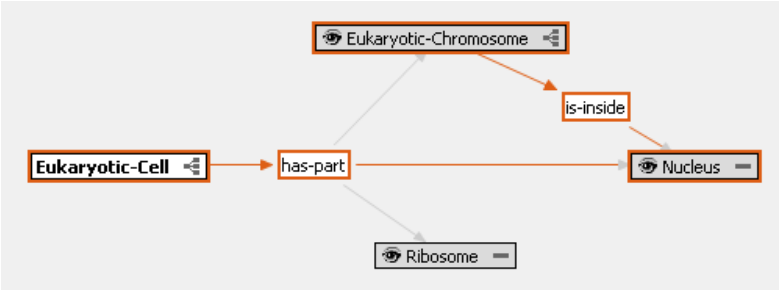
This corresponds to the following first order logic sentence:

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

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

$$\forall\, x : Cell(x) \rightarrow$$
$$hasPart\big(x, f_{Cell}\#1(x)\big) \wedge hasPart\big(x, f_{Cell}\#2(x)\big) \wedge Ribosome\big(f_{Cell}\#1(x)\big)$$
$$\wedge\, Chromosome\big(f_{Cell}\#2(x)\big)$$

The system supports inheritance. Consider the subclass Eukaryotic-Cell which inherits knowledge from Cell. The knowledge ***"Every Eukaryotic-Cell is a Cell. Every Eukaryotic-Cell has part a Eukaryotic-Chromosome, a Ribosome, and a Nucleus, such that the Eukaryotic-Chromsome is inside the Nucleus"*** is modeled using the following graph:



The corresponding Skolemized formula is

$$\forall x : EukaryoticCell(x) \rightarrow Cell(x) \wedge$$
$$hasPart\left(x, f_{ECell}\#1(x)\right) \wedge hasPart\left(x, f_{ECell}\#2(x)\right) \wedge hasPart\left(x, f_{ECell}\#3(x)\right) \wedge$$
$$EukaryoticChromsome\left(f_{ECell}\#3(x)\right) \wedge Nucleus\left(f_{ECell}\#1(x)\right) \wedge Ribosome\left(f_{ECell}\#2(x)\right) \wedge$$
$$isInside(f_{ECell}\#3(x), f_{ECell}\#1(x))$$

Intuitevely, the Chromosome in Eukaryotic-Cell was inherited from Cell, and then specialized into a Eukaryotic-Chromosome. Moreover, the Ribosome was inherited from Cell as well. The Nucleus, was added locally in Eukaryotic-Cell. The advantage of using Skolem functions is that the inheritance can be made explicit by means of equality atoms: if we add $f_{ECell}\#3(x) = f_{Cell}\#2(x)$, $f_{ECell}\#2(x) = f_{Cell}\#1(x)$ to the formula for Eukaryotic-Cell. Doing so makes it clear that the Eukaryotic-Chromsome in Eukaroytic-Cell 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 Eukaryotic-Chromsome in the context of Eukaryotic-Cell as well (in addition to what was modeled for Chromsome itself, of course). Morveover, it is clear then that the Ribosome is the same as the one inherited from Cell:

$$\forall x : EukaryoticCell(x) \rightarrow Cell(x) \wedge$$
$$hasPart\left(x, f_{ECell}\#1(x)\right) \wedge hasPart\left(x, f_{ECell}\#2(x)\right) \wedge hasPart\left(x, f_{ECell}\#3(x)\right) \wedge$$
$$EukaryoticChromsome\left(f_{ECell}\#3(x)\right) \wedge Nucleus\left(f_{ECell}\#1(x)\right) \wedge Ribosome\left(f_{ECell}\#2(x)\right) \wedge$$
$$isInside\left(f_{ECell}\#3(x), f_{ECell}\#1(x)\right) \wedge$$
$$f_{ECell}\#3(x) = f_{Cell}\#2(x) \wedge$$
$$f_{ECell}\#2(x) = f_{Cell}\#1(x)$$

The equalities in the above axiom could not have been asserted if simple existentially quantified variables were used and within Eukaryotic-Chromsome we could not assume that the described Eukaryotic-Chromsome is identical with the Chromsome which gets inherited from Cell.

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 Knowledge Engineering Process Behind KB_Bio_101

The *KB_Bio_101* is created using a systematic knowledge engineering (KE) process developed at SRI. The KE process has the following steps: determine relevant sentences in the textbook, reach consensus on the meaning and identify universal quantifiers, design the representation, encode knowledge using AURA, inspect the knowledge for quality, and then test the knowledge by posing questions. The test questions consist of questions derived from question templates such as: What is X? What is the structure of X? Compare X and Y? Related X to Y? etc. Each chapter is tested using approximately 150 questions. The question output is rated by encoders, teacher biologists, and student biologists. The visual inspection of the KB in our KE process has shown the KB to be nearly 100% accurate. The question-based tests on the KB have shown that

the system answers over 85% of the questions correctly. The questions that do not answer can have a variety of issues that include mismatch with the user expectation, timeouts, or software errors in the system. Our process is extremely thorough and rigorous because of which encoding and testing a chapter can take anywhere between 6-8 person months.

4. The Axiomatic Content of the KB_Bio_101 – An Abstract FOPL Description

We first describe the signature of the KB:

Let *CN* be a set of *class names* (e.g., *Cell*), and *RN* be a set of *relation names* (e.g., *has-part*). Let $AN \subseteq RN$ be a set of *attribute names* (e.g., *color*, *temperature*). In the following, we use $C, C_1, C_2, \ldots\ D, D_1, D_2, \ldots, E, E_1, E_2, \ldots, F, F_1, F_2, \ldots$ to denote classes, and $R, R_1, R_2, \ldots, S, S_1, S_2, \ldots, T, T_1, T_2, \ldots$ to denote relations. *String* denotes a string, e.g. "Cell", and *float* denotes a floating point number (e.g., 22d0). Let $\{x, y, z, x_1, x_2, \ldots\}$ be a set of first-order variables, and, for every $C \in CN$, let $\{ fn_C\#1, fn_C\#2, \ldots \}$ be a set of (Skolem) function symbols.

There are three kinds of attributes:

- Cardinal attribute values.
  For example, "t is 43 years" would be represented as
  $age(t, t_1)$, *the-cardinal-value(t_1,43), cardinal-unit-class(t_1,year).*
- Categorial attribute values.
  For example, "t has color green" would be represented as
  color(t, t_1), the-categorical-value(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), the-scalar-value(t_1,big), scalar-unit-class(t_1,house).

The so-called *value atoms* (the-cardinal-value, cardinal-unit-class, the-categorical-value, the-scalar-value, scalar-unit-class) will be explained below.

We have the following sets of constants:

- the set of scalar constant values: *SCs* = { *small, big,* … }.
- the set of categorical constant values: *CCs* = { *blue, green*, …}.
- the set of cardinal unit classes: *CUCs* = { *meter, year*, … }.
- in addition, the symbols in *CN* and *RN* are constants as well.

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 ∈ *SCs* ∪ *CCs* ∪ *CUCs*, 1 to n type assertions of the form *C(c)*, where *C* ∈ *CN* (the types of the constant).

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

- (CAs) For every class name *C* ∈ *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 Φ[x] is a conjunction of *unary (class) atoms and binary (relation) atoms* over terms $\{ x, fn_C\#1(x), fn_C\#2(x), \dots \}$ - those terms are also called *nodes*, having a free variable *x.*

    There are two special equality relations, namely *equal* and *not-equal*, which are user asserted equality atoms. The intended semantics is the semantics of first-order equality resp. inequality. In order to distinguish them from the equalities in *EQAs*, see below, we use different predicate names here (*equal, not-equal*).

    Moreover, $\Phi[x]$ can contain the following *value atoms*:
    if *t* is a node, *float* is floating point number, and *scalar* ∈ *SCs*, *categorical* ∈ *CCs*, *cardinal-unit-class* ∈ *CUCs*, and *scalar-unit-class* ∈ *CN*, then the following atoms are value atoms:

    - the-cardinal-value(t, float).
    - the-scalar-value(t, scalar).
    - the-categorical-value(t, categorical).
    - cardinal-unit-class(t, cardinal-unit-class).
    - scalar-unit-class(t, scalar-unit-class).

    In addition, the KB contains *qualified number restrictions*. Due to a lack of counting quantifiers, we represent them by means of quadrary atoms of the kind *maxCardinality(t, R, n, C), minCardinality(t, R, n, C), and exactCardinality(t, R, n, C) = maxCardinality(t, R, n, C), minCardinality(t, R, n, C),* where *n* is a non-negative integer, *C* is a class, and *R* is a relation name. The intended semantics of those expressions is the standard description logics semantics [8], but a weaker semantics, i.e. based on counting and closed-world reasoning, might be given to those expressions.

  - (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 $t1 = t2$, where *t1* ∈ $\{ x, x_1, x_2, \dots \}$, and *t2* ∈

$\{x, fn_C\#1(x), fn_C\#2(x), \ldots\}$, linking the variables in the antecedent to the Skolem functions 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 $EQs[x, \ldots]$: $\Theta'[x] = \Theta[x]_{\{t1 \Rightarrow t2,\ t1=t2 \in EQs[x,\ldots]\}}$. Hence, every sufficient condition is also necessary. This is a byproduct of the graphical modeling, as the biologists cannot author sufficient conditions which are not 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 kinds of axioms:

  - (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 relation: $\forall x, y : R(x, y) \rightarrow S(y, x)$
  - (12NAs) 1-to-N cardinality (inverse functionality): $\forall x, y, z : R(x, y) \wedge R(z, y) \rightarrow x = z$
  - (N21As) N-to-1 cardinality (functionality): $\forall x, y, z : R(x, y) \wedge R(x, z) \rightarrow y = z$
  - (TRANSAs) simple transitive closure axioms:
    $\forall x, y, z : R(x, y) \wedge Rstar(y, z) \wedge C(x) \wedge D(y) \wedge E(z) \rightarrow Rstar(x, z)$,
    $Rstar(x, z) = R^*(x, z)$ (a named relation for the transitive closure of R)
  - (GTRANSLAs) generalized transitive closure axioms (left composition):
    $\forall x, y, z : R(x, y) \wedge S(y, z) \wedge C(x) \wedge D(y) \wedge E(z) \rightarrow Rstar(x, z)$,
    $Rstar(x, z) = R^*(x, z)$ (a named relation for the transitive closure of R)
  - (GTRANSRAs) generalized transitive closure axioms (right composition):
    $\forall x, y, z : R(x, y) \wedge S(y, z) \wedge C(x) \wedge D(y) \wedge E(z) \rightarrow Sstar(x, z)$,
    $Sstar(x, z) = S^*(x, z)$ (a named relation for the transitive closure of S)

  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).*

5. The KB_Bio_101 in TPTP FOF Format

The TPTP project defines a standard format for first-order logic, TPTP FOF. The rendering of the KB in TPTP FOF syntax [10,11] is straight forward - every TPTP FOF axiom *ax* is rendered within a surrounding

```
fof(id, axiom,(
  ax )).
```

where *id* is some identifier. The universal quantifiers $\forall x : \ldots.$ and $\forall x, y : \ldots.$ are rendered as `! [ X ] : …` and `! [X, Y] : …`, respectively. The Boolean connectives are rendered as `~`, `&`, `|`, and the implication → as `=>`. Variables are rendered uppercase. Strings are rendered in double quotes.

Skolem functions $f_C\#n$ are rendered as `f_C_n`. Floating point numbers are rendered using the standard scientific notation, e.g., 43.0e0 .

The whole KB is in one big TPTP file, which has the following structure:

1. The *CTAs* are rendered. A fact such as *Size-Constant(big)* is rendered as `Size-Constant(big)`.

   Note that S*ize-Constant* is a class from the KB and hence, axioms may be rendered it as well (e.g., it is a subclass of *Constant, etc.*)

2. Relation axioms are written. For a relation R, all axioms for *R, RAs(R),* are combined into a single conjunction $\Omega[x, y]$ to produce one axiom of the form
   $$\forall\, x, y : R(x, y) \rightarrow\ \Omega[x, y]$$
   which is then rendered in the obvious way.

3. Next, the axioms *DAs(C), TAs(C), EQAs(C)* are combined with the necessary conditions in *NCAs(C)* to produce one axiom of the form
   $$\forall\, x : C(x) \rightarrow\ \Omega[x]$$
   which is then rendered in the obvious way. Note that $\Omega[x]$ is the conjunction of all the atoms in the consequents of the axioms in *DAs(C), TAs(C),* NCAs(C), and *EQAs(C).*

4. Finally, the sufficient axioms from *SCAs(C)* are rendered, in the obvious way, using the syntax described above for necessary conditions.

An excerpt from the KB showing heavily simplified classes for Cell and Eukaryotic-Cell takes the following form in TPTP FOF syntax; also note the documentation in terms of description atoms etc. which we have not discussed. Their meaning should be straight forward:

```
fof(a11860,axiom,(
  ! [X, Y] :
    ( ( has_part(X, Y) )
     =>
      ( tangible_entity(Y) ) ))).

fof(a11861,axiom,(
  ! [X, Y] :
    ( ( has_part(X, Y) )
     =>
      ( tangible_entity(X) ) ))).

fof(a11862,axiom,(
  ( ( has_part(X, Y)
    & has_part(Z, Y) )
   =>
    ( X=Z ) ))).

fof(a11863,axiom,(
  ! [X, Y] :
    ( ( has_part(X, Y) )
     =>
      ( has_structure(X, Y)
        & related_to(X, Y)
        & has_part_or_unit(X, Y)
        & is_part_of(Y, X) ) ))).

fof(a12942,axiom,(
  ! [X, Y, Z] :
```

```
    ( ( has_part_or_unit(X, Y)
      & element(Y, Z)
      & tangible_entity(X)
      & aggregate(Y)
      & tangible_entity(Z) )
     =>
      ( has_part_star(X, Z) ) ))).


fof(a13502,axiom,(
  ! [X] :
    ( ( cell(X) )
     =>
      ( original_name(X, "Cell")
        & description(X, "The basic unit from which living organisms are made, consisting of an
aqueous solution of organic molecules enclosed by a membrane.  All cells arise from existing
cells, usually by a process of division into two.  (Alberts:ECB:G-3).")
        & class2words(X, "cell")
        & living_entity(X)
        & ribosome(fn_cell_1(X))
        & chromosome(fn_cell_2(X))
        & has_part(X, fn_cell_2(X))
        & has_part(X, fn_cell_1(X)) ) ))).


fof(a13504,axiom,(
  ! [X] :
    ( ( eukaryotic_cell(X) )
     =>
      ( original_name(X, "Eukaryotic-Cell")
        & class2words(X, "eukaryotic cell")
        & class2words(X, "eukaryotic-cell")
        & cell(X)
        & nucleus(fn_eukaryotic_cell_1(X))
        & ribosome(fn_eukaryotic_cell_2(X))
        & eukaryotic_chromosome(fn_eukaryotic_cell_3(X))
        & has_part(X, fn_eukaryotic_cell_1(X))
        & is_inside(fn_eukaryotic_cell_3(X), fn_eukaryotic_cell_1(X))
        & has_part(X, fn_eukaryotic_cell_3(X))
        & has_part(X, fn_eukaryotic_cell_2(X))
        & fn_eukaryotic_cell_3(X)=fn_cell_2(X)
        & fn_eukaryotic_cell_2(X)=fn_cell_1(X) ) ))).

fof(a13360,axiom,(
  ! [X] :
    ( ( eukaryotic_chromosome(X) )
     =>
      ( original_name(X, "Eukaryotic-Chromosome")
        & class2words(X, "eukaryotic chromosome")
        & class2words(X, "eukaryotic-chromosome")
        & chromosome(X) ) ))).
```

We have verified the syntax of the KB with the help of the “Local file to upload” functionality under http://www.cs.miami.edu/~tptp/cgi-bin/SystemB4TPTP.

The KB can be downloaded from http://www.ai.sri.com/halo/halobook2010/exported-kb/biokb.html on request (a password and accepting a research use license is required). The KB is 27 MBs big. The overall complexity if witnessed by the following table:

#Classes	#Relations	#Constants	Avg. #Skolems / Class	Avg. #Atoms / NCA	Avg. #Atoms / SCA
6430	455	634	24	64	4

The stats for the class axioms are:

#CTAs	#TAs	#DAs	#EQAs	#Qualified Number Restrictions
714	6993	18616	108755	936

Regarding the relation axioms:

#DRAs	#RRAs	#RHAs	#QRHAs	#IRAs	#12NAs , #N21As	#TRANSAs + #GTRANSLAs + #GTRANSRAs
449	447	13	39	212	10, 132	431

5. Summary

An initial version of the *KB_Bio_101* is now available, and we propose to actively engage with the research community to first define an acceptable representation, and then participate in an experimental evaluation of the results of the reasoning tasks suggested below

Interesting reasoning tasks are, for example, computation of similarity and differences between classes, and computation of relationships between classes:

- In a similarity reasoning task, given classes A and B, the task is to compute the intersection and difference between their properties. The similarity computation is similar to the least common subsumer operation in description logics [8]. In our system, similarities are aligned based on semantic and syntactic similarity notions, similar to heuristics found in ontology alignment algorithms.
- In the relationship reasoning task, we first instantiate each class in the KB, and then, given two instances of classes A and B, we wish to compute all possible paths of a certain length between those instances. This is a computationally explosive task and our current system relies on an extensive set of heuristics and depth bounds to achieve an acceptable performance.

In general, reasoning with the KB is likely to be undecidable, because of cycles and function symbols in (implication) consequents of axioms.

Acknowledgment

This work has been funded by Vulcan Inc.

References

1. Project Halo. http://www.projecthalo.com
2. David Gunning, Vinay K. Chaudhri, Peter Clark, Ken Barker, Shaw-Yi Chaw, Mark Greaves, Benjamin Grosof, Alice Leung, David McDonald, Sunil Mishra, John Pacheco, Bruce Porter, Aaron Spaulding, Dan Tecuci, and Jing Tien. *Project Halo Update — Progress Toward Digital Aristotle.* AI Magazine, Fall 2010.
3. Jane B. Reece, Lisa A. Urry, Michael L. Cain, Steven A. Wasserman, Peter V. Minorsky, and Robert B. Jackson. *Campbell Biology*, 9th ed. Benjamin Cummings, 2011.
4. *Inquire: An Intelligent Textbook.* http://aivideo.org/2012/
5. Peter E. Clark and Bruce Porter. *Knowledge Machine Users's Guide*. Technical Report, University of Texas at Austin.
6. Melvin Fitting. *First-Order Logic and Automated Theorem Proving.* Springer, 1996.
7. Benjamin N. Grosof. *SILK: Higher Level Rules with Defaults and Semantic Scalability.* In Axel Polleres and Terrance Swift, editors, Web Reasoning and Rule Systems, Third International Conference, RR 2009, Volume 5837 of Lecture Notes in Computer Science, Springer, 2009.
8. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. 2nd Edition. Cambridge University Press, 2007.
9. M. Gelfond and V. Lifschitz. Logic *Programs with Classical Negation.* In D.Warren and Peter Szeredi, editors, Logic Programming: Proceedings of the Seventh International Conference, 1990.
10. Geoff Sutcliffe. The *TPTP World-Infrastructure for Automated Reasoning.* In Edmund M. Clarke, Andrei Voronkov, editors. Logic for Programming, Artificial Intelligence, and Reasoning - 16th International Conference, LPAR-16, 2010
11. *TPTP FOF Syntax:* Online http://www.cs.miami.edu/~tptp/TPTP/SyntaxBNF.html#fof_formula