Improving the Competency of First-Order Ontologies

Javier Álvez
LoRea Group
University of the Basque
Country UPV/EHU
javier.alvez@ehu.eus

Paqui Lucio
LoRea Group
University of the Basque
Country UPV/EHU
paqui.lucio@ehu.eus

German Rigau
IXA NLP Group
University of the Basque
Country UPV/EHU
german.rigau@ehu.eus

ABSTRACT

We introduce a new framework to evaluate and improve first-order (FO) ontologies using automated theorem provers (ATPs) on the basis of competency questions (CQs). Our framework includes both the adaptation of a methodology for evaluating ontologies to the framework of first-order logic and a new set of non-trivial CQs designed to evaluate FO versions of SUMO, which significantly extends the very small set of CQs proposed in the literature. Most of these new CQs have been automatically generated from a small set of patterns and the mapping of WordNet to SUMO. Applying our framework, we demonstrate that Adimen-SUMO v2.2 outperforms TPTP-SUMO. In addition, using the feedback provided by ATPs we have set an improved version of Adimen-SUMO (v2.4). This new version outperforms the previous ones in terms of competency. For instance, "*Humans can reason*" is automatically inferred from Adimen-SUMO v2.4, while it is neither deducible from TPTP-SUMO nor Adimen-SUMO v2.2.

Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods

General Terms

Experimentation

1. INTRODUCTION

Ontologies are being used in a wide range of applications and knowledge based systems [2]. Like any other component of a system, an ontology requires a repetitive process of refinement and evaluation during its development and application lifecycle. Ontologies can be evaluated by considering their use in an application when performing correct predictions on inferencing [25]. In order to enable better reasoning capabilities, the inferencing process should be able to deduce from the ontology as much correct implicit knowledge

as possible. In [11], the authors propose to use a set of competency questions (CQs) to evaluate an ontology, which are goals that the ontology is expected to answer. The proposed methodology can be applied to any formal ontology if there exists a decision algorithm for the underlying logic.

In general the process of obtaining CQs is not automatic but creative [8]. Depending on the size and complexity of the ontology, the process of creating a suitable set of CQs is by itself a very challenging and costly task.

Although OWL-DL [14] is currently one of the most common formal knowledge representation formalisms, it is unable to cope with expressive ontologies like Cyc [20], DOLCE [9] or SUMO [21]. Fortunately, state-of-the-art automatic theorem provers (ATPs) for first-order logic (FOL) like Vampire [28] or E [29] are highly sophisticated systems that have been proved to provide advanced reasoning support to substantial FOL conversions of expressive ontologies, including first-orderized Cyc [26], SUMO [15], TPTP-SUMO [24] and Adimen-SUMO [1]. Despite these preliminary experiments, as far as we know, there is no previous work that applies the methodology of [11] to first-order (FO) ontologies using FOL ATPs.

The contributions of this paper are manyfold. First, following [11], we present a new framework to evaluate and improve FO ontologies using ATPs. Second, we introduce a new set of very large and non-trivial CQs designed to evaluate FO versions of SUMO. Our set includes 64 creative CQs for development and more than 7,000 automatically generated CQs for testing. Our set of creative CQs extends the 33 questions from the CSR (Common Sense Reasoning) problem domain of TPTP (Thousands of Problems for Theorem Provers) [30] and the 5 questions described in [1]. Additionally, we have also devised and developed a novel automatic procedure for generating a very large set of non-trivial CQs from a small set of conceptual patterns on the basis of the knowledge encoded in WordNet (WN) [7] and its mapping to SUMO [22]. Third, as a result of the application of our framework, we create a new version of Adimen-SUMO. Finally, using our new framework and CQs, we carry out an empirical comparison of the existing FO versions of SUMO. According to our experimental results, the new version of Adimen-SUMO outperforms TPTP-SUMO and all previous versions of Adimen-SUMO. For example, from Adimen-SUMO v2.4 ATPs infer that "*Tables do not have a brain*" and "*Humans can reason*", but "*Organisms cannot be dead*" and "*Tables can eat*" are not inferred. However, ATPs yield the opposite results from TPTP-SUMO and Adimen-SUMO v2.2 in the four cases. Both the new version of Adimen-

Table 1: Some figures about SUMO, TPTP-SUMO and Adimen-SUMO

	SUMO	TPTP-SUMO	Adimen-SUMO
Objects	20,081	2,920	1,009
Classes	5,563	2,086	2,124
Relations	369	208	208
Attributes	2,153	68	66
Total	**28,166**	**5,282**	**3,407**

SUMO and the new set of CQs are freely available.[1]

Obviously, this type of non-trivial inferences could be very useful for a wide range of knowledge intensive applications. For instance, to help validating the consistency of associated semantic resources like WN, or to derive new explicit knowledge from them. Furthermore, WN is being used world-wide to anchor different types of semantic resources and wordnets in many languages.[2] Therefore, similar inferences can be obtained for other semantic resources and languages other than English. Likewise, WN is connected to several databases such as OpenCyc [27], DBpedia [3, 6] or YAGO [13] thanks to the Linked Open Data (LOD) cloud initiative [5]. The interlinking of these diverse databases to a fully operational upper level ontology promises a "Web of Data" that will enable machines to more easily exploit its content [16].

In the next two sections, we first introduce SUMO and its FOL versions, and then our adaptation of the methodology proposed by [11]. In Section 4, we illustrate the process of improving an ontology, which yields Adimen-SUMO v2.4, by providing some examples. Next, the process of automatically obtaining a new set of CQs from WN and its mapping to SUMO is described in Section 5. In Section 6, we report on the competency of TPTP-SUMO and the different versions of Adimen-SUMO. In the last two sections, we respectively provide some concluding remarks for discussion and our future research lines.

2. FIRST-ORDER VERSIONS OF SUMO

SUMO[3] [21] has its origins in the nineties, when a group of engineers from the IEEE Standard Upper Ontology Working Group pushed for a formal ontology standard. Their goal was to develop a standard upper ontology to promote data interoperability, information search and retrieval, automated inference and natural language processing.

SUMO is expressed in SUO-KIF (Standard Upper Ontology Knowledge Interchange Format [23]), which is a dialect of KIF (Knowledge Interchange Format [10]). Both KIF and SUO-KIF can be used to write FOL formulas, but its syntax goes beyond FOL. Consequently, SUMO cannot be directly used by FOL ATPs without a suitable transformation [1].

There exist different proposals for converting large portions of SUMO into a FO ontology. In [24], the authors report some preliminary experimental results evaluating the query timeout for different options when translating SUMO into FOL. Evolved versions of the translation described in [24] can be found in the TPTP Library[4] (hereinafter TPTP-SUMO). In [1], we use ATPs for reengineering around 88% of SUMO, obtaining Adimen-SUMO. Both TPTP-SUMO and

Adimen-SUMO inherits information from the top and the middle levels of SUMO (from now on, the *core* of SUMO), thus discarding all the information from the domain ontologies. In Table 1, we provide some figures comparing the explicit content of SUMO, TPTP-SUMO and Adimen-SUMO. It is easy to see that the most significant difference between TPTP-SUMO and Adimen-SUMO is the number of objects, which is due to the fact that TPTP-SUMO introduces many instances that should be inferred from the knowledge of the ontology.

An example of the practical inference capabilities of TPTP-SUMO and Adimen-SUMO is the CQ *"Boys are not domestic animals"*

$$(=> \tag{1}$$
$$(\text{instance ?OBJ Boy})$$
$$(\textbf{not}$$
$$(\text{instance ?OBJ DomesticAnimal})))$$

that can be proved using ATPs.[5] In the next sections, we will also provide some CQs that cannot be inferred from neither TPTP-SUMO nor Adimen-SUMO.

3. COMPETENCY QUESTIONS AND FIRST-ORDER ONTOLOGIES

In this section, we describe how to adapt the methodology introduced in [11] for evaluating and improving large and complex FO ontologies using state-of-the-art ATPs like Vampire [28] or E [29].

In [11], the authors propose to evaluate the expressiveness of an ontology by proving completeness theorems w.r.t. a set of CQs. The proof of completeness theorems requires to check whether a given CQ is entailed by the ontology or not. For this purpose, we use Vampire v3.0,[6] which works by refutation within a given execution-time limit. Similarly, we could also use E or other ATPs that work by refutation. Theoretically, if the conjecture is entailed by the ontology, then ATPs will eventually find a refutation given enough time (and space). However, theorem proving in FOL is a very hard problem, so it is not reasonable to expect ATPs to find a proof for every entailed conjecture [18]. Thus, if the ATP is able to find a prove for a conjecture, then we know for sure that the corresponding CQ is entailed by the ontology. However, if the ATP cannot find a proof, we do not know if (a) the conjecture is not entailed by the ontology or (b) although the conjecture is entailed, the ATP has not been able to find the proof within the provided execution-time limit. Due to the semi-decidability problem of FOL, increasing the execution-time limit is not a solution for con-

---

jectures that are not entailed. For the same reason, using other systems that do not work by refutation (for example, by model generation) is not a general solution. To overcome this problem, we consider three possibilities when testing the ontology w.r.t. a given CQ: the test may be (i) *passing*, (ii) *non-passing* or (iii) *unknown*, as we next describe.

As proposed in [11], our method is based on a set of CQs written as conjectures in the language of the ontology. The set of CQs is partitioned into two classes: *truth-tests* and *falsity-tests*, depending on whether we expect the conjecture to be entailed by the ontology or not. For example, let us consider the CQs *"Men cannot be pregnant"* and *"Organisms cannot be dead"*:

$$
\begin{aligned}
&(\Rightarrow \\
&\quad (\textbf{and} \\
&\qquad (\text{instance ?HUMAN Human}) \\
&\qquad (\text{attribute ?HUMAN Pregnant})) \\
&\quad (\textbf{not} \\
&\qquad (\text{instance ?HUMAN Man}))) \\
\end{aligned} \tag{2}
$$

$$
\begin{aligned}
&(\Rightarrow \\
&\quad (\text{instance ?ORG Organism}) \\
&\quad (\textbf{not} \\
&\qquad (\text{attribute ?ORG Dead}))) \\
\end{aligned} \tag{3}
$$

According to the common sense knowledge, the first CQ is a truth-test, whereas the second one is a falsity-test (since *"Organisms can be dead"*). Following this division, our method proceeds in two steps. In the first step, we deal with the set of truth-tests as conjectures. A truth-test is classified as *passing* if the ATP proves that the corresponding conjecture is entailed by the ontology, and it would be classified as *non-passing* if the ATP could prove that the conjecture is not entailed. However, as discussed above, when no proof is found we do not know whether the conjecture is entailed or not, thus we classify the truth-test as *unknown*. In the second step, we deal with the set of falsity-tests, which are supposed not to be inferred from the ontology. Hence, we classify a falsity-test as *non-passing* when the ATP proves that corresponding conjecture is entailed by the ontology, and as *unknown* when the ATP does not find any proof.

In practice, unknown truth-tests are treated as non-passing, since ATPs have not been able to prove that the corresponding conjectures are entailed by the ontology. On the contrary, unknown falsity-tests are treated as passing. When a test is classified as non-passing (or unknown in the case of truth-tests), we proceed to correct the ontology. The correction is creative and directed by the problem at hand. In the case of truth-tests, we do not obtain any information for the tests that are not classified as *passing* (since ATPs find no proof), thus the correction is harder and even we do not know if any correction is required. With respect to falsity-tests, the proof generated by ATPs is used to isolate the controversial axioms, performing additional tests when necessary. The feedback provided by ATPs is very helpful for detecting modelling errors. As ATPs are continuously evolving, our method is producing more precise and useful outcomes, but the semi-decidability problem of FOL still remains. In frameworks where the underlying logic is decidable (like OWL-DL), our framework would be also applicable with the advantage of becoming exact.

In order to carry out the experiments described in this paper, we proceed as follows. First, we collect all creative queries available from the literature. In particular, the 33

questions from the CSR (Common Sense Reasoning) problem domain of TPTP (Thousands of Problems for Theorem Provers) [30] and the 5 questions described in [1]. Then, we extend this reduced set of 38 CQs with 26 new creative CQs. All these 64 creative CQs have been classified manually as 50 *truth-tests* (the 38 old plus 12 new CQs) and 14 *falsity-tests* (all new). When applying our framework, we use this reduced set of 64 creative CQs as a dataset for development (as explained in Section 4). For improving the competency of the ontology, we only consider the results and traces returned by the ATPs when proving the CQs of this reduced set of creative tests. Second, for testing the competency of new FO versions of SUMO, we automatically create a very large set of CQs derived from WN and its mapping to SUMO (as described in Section 5). In our framework, we use this large set of more than 7,000 automatic CQs as a dataset for testing (see Section 6).

4. IMPROVING A FIRST-ORDER ONTOLOGY

In this section, we report on the experience of improving an FO ontology applying our framework and the set of 64 creative CQs. In particular, we provide some examples of truth- and falsity-tests, explaining how we have used them and the feedback provided by ATPs for improving our ontology. As a result of this process, we have derived Adimen-SUMO v2.4.

As discussed in the above section, we have to improve the ontology when the conjecture corresponding to a truth-test is not proven to be entailed, or when ATPs prove that the conjecture corresponding to a falsity-test is entailed. In the former case, we get no more feedback from ATPs, since no proof is found. Hence, we have to manually check the ontology to search for the modelling error. In the latter case, we obtain a proof, which includes the incorrect axioms. Roughly speaking, the modelling error can refer to ontological concepts (relations or objects) that are used to define the structure and main features of the ontology itself —the so-called *basic* concepts— or to concepts that serves to describe the knowledge that is contained in the ontology —from now on, *non-basic* concepts—. For example, *instance*, *subclass*, *disjoint*, *partition*, *attribute*, *subAttribute*, *contraryAttribute* and *exhaustiveAttribute* are basic concepts in Adimen-SUMO, whereas *NullList* and *Animal* are examples of non-basic concepts. According to our experience, there are three typical types of errors than can be informally described as follows:

- Missing characterization: the modelling error refers to a non-basic concept C, but the ontology lacks the axiomatization of C. This is the simplest case, since the solution consists in axiomatizing C.

- Too weak characterization: the modelling error refers to a non-basic concept C that is characterized, but in a way too weak. In this case, the solution consists in repairing the characterization (updating one or more axioms) of C.

- Unsuitable characterization of basic concepts: the modelling error refers to a non-basic concept C which is well-characterized in terms of some basic concept B, but B is not suitable defined. This is the most complex

case, since the solution may require modifying several basic definitions of the ontology, or even its structure.

However, the distinction between basic and non-basic concepts in an ontology is not always clearly stated and is usually dependent upon one's interpretation. This is the case of SUMO, where all concepts are defined in terms of SUMO itself, preventing a direct translation of SUMO into FOL [1]. This problem also prevents a more formal characterization of modelling errors. Next, we provide some examples of the modelling errors described above and the CQs that have enabled its automated detection by following our proposal.

Regarding truth-tests, let us consider the CQs *"A list containing at least one item is not empty (null)"* (4), *"Tables do not have a brain"* (5) and *"Men cannot be pregnant"* (2):

```
(=>                                          (4)
  (and
    (instance ?LIST List)
    (instance ?ITEM Entity)
    (inList ?ITEM ?LIST))
  (not
    (equal ?LIST NullList)))
```

```
(=>                                          (5)
  (and
    (instance ?BRAIN Brain)
    (instance ?TABLE Table))
  (not
    (properPart ?BRAIN ?TABLE)))
```

After some initial experiments with ATPs, we realize that these conjectures are entailed from neither Adimen-SUMO nor TPTP-SUMO. Analysing the knowledge required to answer each question, we find the following problems.

In the case of the CQ *"A list containing at least one item is not empty (null)"*, the object *NullList* is only axiomatized to be instance of *List*, hence a proper characterization is missing. To solve this problem, we include the following axiom in Adimen-SUMO v2.4 as characterization of *NullList*:

```
(not                                         (6)
  (inList ?ITEM NullList))
```

Including this axiom, Adimen-SUMO v2.4 entails *"A list containing at least one item is not empty (null)"* (4).

The second CQ *"Tables do not have a brain"* (5) cannot be proved because a too weak characterization of the concept *Animal*. In particular, the source of the problem is:

```
(=>                                          (7)
  (and
    (instance ?STRUCTURE AnimalAnatomicalStructure)
    (instance ?ANIMAL Organism)
    (part ?STRUCTURE ?ANIMAL))
  (instance ?ANIMAL Animal))
```

This axiom can only be applied to instances of *Organism*, which is not the case of *Table*. However, it is not necessary to restrict the use of this axiom to instances of *Organism*, since the ontology already entails that only *Animals* can have *AnimalAnatomicalStructures*. Thus, we could relax the antecedent of the formula by simply removing that restriction. However, *part* is defined as *PartialOrderingRelation*, hence it is reflexive, and the classes *AnimalAnatomicalStructure* and *Animal* are defined to be disjoint. Thus, from the resulting axiom it would be possible to infer that any instance of

AnimalAnatomicalStructure, which is trivially part of itself, is also instance of *Animal*, contradicting the disjointness of these classes. Therefore, it is more suitable the use of *properPart* (which is irreflexive) in the resulting axiom instead of *part*. To sum up, the above axiom (7) is rewritten to:

```
(=>                                          (8)
  (and
    (instance ?STRUCTURE AnimalAnatomicalStructure)
    (properPart ?STRUCTURE ?ANIMAL))
  (instance ?ANIMAL Animal))
```

Adimen-SUMO v2.4 entails *"Tables do not have a brain"* by means of replacing the old axiom (7) with (8).

The problem of non-passing the truth-test *"Men cannot be pregnant"* (2) comes from an unsuitable characterization of some basic relations about attributes. More specifically, *contraryAttribute* and *exhaustiveAttribute* are variable arity relations that constrain the use of attributes. As in the case of the other variable arity relations defined in SUMO (such as *partition*, *disjointDecomposition* or *exhaustiveDecomposition*), *contraryAttribute* and *exhaustiveAttribute* are characterized on the basis of *inList* and *ListOrderFn*. In Adimen-SUMO, the so-called *row operators* are used for a proper characterization [1] (this problem remains unsolved in TPTP-SUMO). However, Adimen-SUMO v2.2 directly inherited from SUMO unsuitable characterizations of *inList* and *ListOrderFn* that prevent to prove most of the truth-tests involving attributes. To fix this problem, we have included in Adimen-SUMO v2.4 two axioms characterizing *contraryAttribute* and *exhaustiveAttribute*, which enables Adimen-SUMO v2.4 to entail *"Men cannot be pregnant"*.

Now, let us consider the falsity-tests *"Tables can be living"* (9) and *"Organisms cannot be dead"* (3):

```
(exists (?TABLE)                             (9)
  (and
    (instance ?TABLE Table)
    (attribute ?TABLE Living)))
```

We have performed many runs using different ATPs and none of them finds a proof of the goals (9) and (3) from Adimen-SUMO v2.2 or TPTP-SUMO. Likewise, ATPs do not find either any refutation for (9) from Adimen-SUMO v2.4. However, the new characterization of *contraryAttribute* and *exhaustiveAttribute* in Adimen-SUMO v2.4 enables ATPs to find a proof of *"Organisms cannot be dead"* (3). From this proof, we discover that the problem is related with the following axioms:

```
(contraryAttribute Dead Living)              (10)
```

```
(subAttribute Dead Unconscious)              (11)
```

```
(instance Unconscious ConsciousnessAttribute) (12)
```

```
(<=>                                         (13)
  (and
    (instance ?AGENT SentientAgent)
    (attribute ?AGENT Living))
  (exists (?ATTR)
    (and
      (instance ?ATTR ConsciousnessAttribute)
      (attribute ?AGENT ?ATTR))))
```

According to axiom (10), it is not possible to have both *Living* and *Dead* as attribute. However, by (11) and (12), *Dead* is an instance of *ConciousnessAttribute*. Moreover,

having *Dead* as attribute implies to also have *Living* by (13). Analysing this set of conflictive axioms, we have decided to remove axiom (11), which defines *Dead* as subattribute of *Unconscious*, from Adimen-SUMO v2.4. This incorrectness was hidden in both Adimen-SUMO 2.2 and TPTP-SUMO due to the inappropriate characterization of attributes.

After this development phase, Adimen-SUMO v2.4 passes the 50 creative truth-tests, whereas 23 and 15 creative truth-tests are classified as unknown by TPTP-SUMO and Adimen-SUMO v2.2 respectively. Regarding the creative falsity-tests, all of them are classified as unknown by the three ontologies.

5. DERIVING COMPETENCY QUESTIONS FROM WORDNET

In order to build our benchmark, we have used the mapping from WN to SUMO [22]. This mapping connects each synset of WN into a term of SUMO using three relations: *equivalence*, *subsumption* and *instance*. These relations will be denoted by concatenating the symbols '=' (*equivalence*), '+' (*subsumption*) and '@' (*instance*) to the corresponding SUMO concept. For example, $piloting_n^2$, $education_n^4$ and $zero_a^1$ are connected to *Pilot=*, *EducationalProcess+* and *Integer@*. Additionally, the complementary of the relations *equivalence* and *subsumption* are also used.

The mapping from WN to SUMO uses terms from the core of SUMO, but also from the domain ontologies. However, both TPTP-SUMO and Adimen-SUMO only use axioms from the core of SUMO. Thus, our first task has been to obtain a mapping from WN to the core of SUMO on the basis of the mapping from WN to SUMO. To this end, for each WN synset not mapped to a term covered by both TPTP-SUMO and Adimen-SUMO, we have conveniently used the structural relations of SUMO (*instance*, *subclass*, *subrelation* and *subAttribute*) to inherit the term of the core of SUMO to which the synset is connected. Note that this process sometimes requires to modify the mapping relation. For example, the synset $frying_n^1$ is connected to the SUMO class *Frying=*, which belongs to the domain ontology *Food*. In the same domain ontology, *Frying* is defined to be subclass of *Cooking*, which is defined in the top level of SUMO. Consequently, *Frying* is not defined in the core of SUMO, but *Cooking* is. Thus, the synset $frying_n^1$ can be connected to *Cooking* in the resulting mapping. However, instead of *equivalence*, $frying_n^1$ is connected to *Cooking* by the *subsumption* mapping relation: that is, *Cooking+*. The total number of mappings to the core of SUMO (114,948) is slightly smaller than the number of mappings to SUMO (115,872) since some terms are not properly defined. This is mainly due to the fact that some terms in the mapping derived from older versions of SUMO are not longer available in the current one. For example, the synsets $salmon_n^1$ and $architect_n^2$ are respectively connected to the SUMO concepts *Salmon=* and *Architect=*, which do not appear in the latest versions of SUMO.

After obtained a suitable mapping from WN to the ontologies that we want to compare, we have designed several conceptual patterns of questions regarding the information about antonyms and processes in WN.

5.1 Antonym patterns

WN provides a set of 8,689 antonym-pairs, including nouns, verbs, adjectives and adverbs, from which 7,410 antonym-

pairs can be properly mapped to the core of SUMO, as described above. However, we only consider the antonym-pairs where both synsets are connected using the *equivalence* mapping relation (in total, 190 pairs), discarding those where the *subsumption* mapping relation is used. For these 190 antonym-pairs, we propose two conceptual patterns of questions. The first pattern is based on the fact that two SUMO classes connected to antonym synsets of WN cannot have common instances. For example, the antonym synsets $frozen_n^1$ and $liquescent_n^1$ are respectively connected to *Freezing=* and *Melting=*. Thus, from the above antonym-pair, we derive the following competency question:

$$
\begin{aligned}
&(\textbf{not} \qquad\qquad\qquad\qquad\qquad\qquad\qquad (14)\\
&\quad (\textbf{exists } (?\mathtt{X})\\
&\quad\;\; (\textbf{and}\\
&\qquad (\text{instance } ?\mathtt{X} \text{ Melting})\\
&\qquad (\text{instance } ?\mathtt{X} \text{ Freezing}))))
\end{aligned}
$$

Similarly, the second conceptual pattern states that two attributes connected to antonym synsets are not compatible. For example, from the antonym synsets $waking_n^1$ and $sleeping_n^1$, which are connected to *Awake=* and *Asleep=*, we derive the following competency question:

$$
\begin{aligned}
&(\textbf{not} \qquad\qquad\qquad\qquad\qquad\qquad\qquad (15)\\
&\quad (\textbf{exists } (?\mathtt{X})\\
&\quad\;\; (\textbf{and}\\
&\qquad (\text{attribute } ?\mathtt{X} \text{ Awake})\\
&\qquad (\text{attribute } ?\mathtt{X} \text{ Asleep}))))
\end{aligned}
$$

Applying these two patterns on the 190 antonym-pairs where both synsets are connected used *equivalence*, we obtain 64 different truth-tests. By negating each of the above 64 CQs, we also obtain 64 different falsity-tests.

5.2 Process patterns

Regarding processes, we have used the information in the morphosemantic database,[7] which contains semantic relations between morphologically related nouns and verbs. From the 14 semantic relations defined in the database, we select *agent*, *result*, *instrument* and *event*. The first three ones relate a process (verb) which its corresponding agent / result / instrument (noun), from which we infer 1,280 CQs by simply stating the same property in terms of SUMO. For example, WN establishes that the *result* of $compose_v^2$ is a $composition_n^4$, which are respectively mapped to *ComposingMusic+* and *MusicalComposition=*:

$$
\begin{aligned}
&(\textbf{exists } (?\mathtt{X} \; ?\mathtt{Y}) \qquad\qquad\qquad\qquad\qquad (16)\\
&\quad (\textbf{and}\\
&\qquad (\text{instance } ?\mathtt{X} \text{ ComposingMusic})\\
&\qquad (\text{result } ?\mathtt{X} \; ?\mathtt{Y})\\
&\qquad (\text{instance } ?\mathtt{Y} \text{ MusicalComposition})))
\end{aligned}
$$

As before, we also obtain 1,280 falsity-tests by negating the previous ones.

The last relation *event* connects nouns and verbs referring to the same process. Being the same process, we assume that both the noun and the verb should be mapped to the same class of SUMO. Thus, if the noun and the verb are mapped to different SUMO class constants, our hypothesis is that the mapping is wrong. Following this criterion,

---

[7]Available at `http://wordnetcode.princeton.edu/standoff-files/morphosemantic-links.xls`.

Table 2: Evaluation of SUMO-based FO ontologies

Tests	TPTP-SUMO					Adimen-SUMO v2.2					Adimen-SUMO v2.4				
	P	N	U	t(P)	t(N)	P	N	U	t(P)	t(N)	P	N	U	t(P)	t(N)
Truth-tests (3,556)	4	–	3,552	361.97 s.	–	89	–	3,467	56.06 s.	–	894	–	2,662	56.46 s.	–
Antonym pattern (64)	3	–	61	473.26 s.	–	17	–	47	6.77 s.	–	45	–	19	32.18 s.	–
Relation pattern (1,280)	0	–	1,280	–	–	11	–	1,269	121.09 s.	–	176	–	1,104	63.72 s.	–
Event pattern #1 (25)	0	–	25	–	–	2	–	23	18.72 s.	–	7	–	18	108.30 s.	–
Event pattern #2 (330)	0	–	330	–	–	26	–	304	45.46 s.	–	115	–	215	44.16 s.	–
Event pattern #3 (1,857)	1	–	1,756	28.13 s.	–	33	–	1,824	70.40 s.	–	551	–	1,306	58.03 s.	–
Falsity-tests (3,556)	–	466	3,090	–	25.19 s.	–	493	3,063	–	6.89 s.	–	487	3,069	–	6.88 s.
Antonym pattern (64)	–	4	60	–	22.16 s.	–	2	62	–	3.31 s.	–	5	59	–	14.06 s.
Relation pattern (1,280)	–	4	1,276	–	191.93 s.	–	31	1,249	–	97.69 s.	–	22	1,258	–	114.60 s.
Event pattern #1 (25)	–	0	25	–	–	–	0	25	–	–	–	0	25	–	–
Event pattern #2 (330)	–	71	259	–	23.73 s.	–	72	258	–	0.57 s.	–	72	258	–	1.18 s.
Event pattern #3 (1,857)	–	387	1,470	–	23.76 s.	–	388	1,469	–	0.82 s.	–	388	1,469	–	1.73 s.

we propose different conceptual patterns of questions, depending on the used mapping relations, with the purpose of detecting wrong-mappings. If both synsets are connected to two different SUMO class constants using the *equivalence* mapping relation, it should be possible to prove that the two class constants denote different classes. For example, $kill_v^{10}$ and $killing_n^2$ are respectively connected to the SUMO classes *Death=* and *Killing=*, hence we derive the following CQ:

$$\text{(not} \tag{17}$$
$$\quad \text{(equal Death Killing))}$$

Using this pattern, we derive 25 CQs. The second pattern of question focuses on the case where the synsets are connected to different SUMO class constants and using different mapping relations. That is, one synset is connected using the *equivalence* mapping relation, whereas the other synset is connected using *subsumption*. Being the mapping information less precise than in the first case, it does not suffice to prove that the classes are different. In this case, the pattern states that the class connected using *equivalence* cannot be subclass of the class connected using *subsumption*. For example, *event* relates $repair_n^1$, which is connected to *Repairing=*, and $repair_v^1$, which is connected to *Pretending+*. Therefore, we assume that *Repairing* cannot be subclass of *Pretending*, deriving the following CQ:

$$\text{(not} \tag{18}$$
$$\quad \text{(subclass Repairing Pretending))}$$

From the second *event* pattern of questions, we derive 330 CQs. In fact, this second pattern can be seen as a particular case of the third one, where both synsets are connected using the *subsumption* mapping relation. In this case, the pattern states that none of the connected SUMO classes can be subclass of the other one. For example, *event* relates the synsets $measure_v^4$ and $appraisal_n^1$, which are respectively connected to *Judging+* and *Comparing+*. Consequently, we derive the following CQ:

$$\text{(not} \tag{19}$$
$$\quad \text{(or}$$
$$\quad\quad \text{(subclass Judging Comparing)}$$
$$\quad\quad \text{(subclass Comparing Judging)))}$$

Using this third pattern, we obtain 1,857 CQs.

In total, we obtain 2,212 truth-tests by stating that the mapping is not correct, and the corresponding 2,212 falsity-tests stating that the mapping is correct.

6. EVALUATING FIRST-ORDER ONTOLOGIES

In this section, we summarize the evaluation results of TPTP-SUMO and the different versions of Adimen-SUMO using the methodology proposed in Section 3. For this evaluation, we have used the set of 7,112 CQs that have been automatically obtained from WN, as described in the above section.

Table 2 sums up some runtime figures of the ATP Vampire 3.0[8] [28] when evaluating TPTP-SUMO and Adimen-SUMO with an execution time limit of 600 seconds.[9] For each ontology, we provide the number of passing (*P* column), non-passing (*N* column) and unknown CQs (*U* column), together with the average runtimes of passing (*t(P)* column) and non-passing (*t(N)* column) CQs. It is worth to remark that the average runtime of the CQs classified as unknown is the maximum execution time (600 seconds), since no proof is found. From the results, it is clear that Adimen-SUMO v2.4 outperforms Adimen-SUMO v2.2 in terms of competency in both the truth-test (more passing tests) and the falsity-test category (less non-passing tests). Regarding efficiency, the average runtime of Adimen-SUMO v2.4 is longer since the passed additional tests require more complex proofs. Similarly, Adimen-SUMO v2.2 outperforms TPTP-SUMO in the truth-test category, since TPTP-SUMO only passes 4 truth-tests while Adimen-SUMO v2.2 passes 89. Regarding falsity-tests, TPTP-SUMO is the ontology with less non-passing tests, but the average runtime is clearly longer. Thus, we think that the number of non-passing falsity-tests of TPTP-SUMO would be larger if we used a longer execution time limit for the experimentation.

Recall that non-passing falsity-tests can provide useful information to improve the ontology, as in the case of the CQ *"Organisms cannot be dead"* (see Section 4). Additionally, non-passing falsity-tests also provide very useful information. For example, the verbs $whisper_v^1$ and $shout_v^1$, which are antonyms in WN, are mapped to the SUMO classes *Speaking=* and *Vocalizing=* respectively. Being antonyms, we expect that these two SUMO classes, *Speaking* and *Vocalizing*, do not have any common instance, as stated by the next conjecture which corresponds to a CQ included in the

---

[8] http://www.vprover.org
[9] In this experimentation, we have used a standard 64-bit Intel® Core™ i7-2600 CPU @ 3.40GHz desktop machine with 16GB of RAM.

automatic falsity-test set:

$$\begin{aligned}
&(\textbf{exists } (?\texttt{X}) \qquad\qquad\qquad\qquad (20)\\
&\quad (\textbf{and}\\
&\qquad (\text{instance }?\texttt{X Vocalizing})\\
&\qquad (\text{instance }?\texttt{X Speaking})))
\end{aligned}$$

However, ATPs can infer the above conjecture from Adimen-SUMO v2.4. That is, Adimen-SUMO v2.4 does not pass this falsity-test. This fact serves to detect that the mapping of the verbs $whisper_v^1$ and $shout_v^1$ to SUMO is not suitable. On the contrary, the CQ is classified as unknown when evaluating TPTP-SUMO and Adimen-SUMO v2.2, which prevents to detect the incorrect mapping.

7. DISCUSSION

We have shown a new framework and experimental results for evaluating and improving large and complex FO ontologies using ATPs. Our results show the appropriateness of using ATPs in debugging FO ontologies, as well as their practical use for inferring non-trivial statements.

Although the authors of [4] claim for the necessity of proving the formal faithfulness of a logical translation of SUMO, the formats KIF and SUO-KIF lack a formal notion of logical consequence and a formal deduction system,[10] where that kind of mathematical results (such as conservative extension) relies on. Thus, following [1], our repairs just intend to improve the reasoning capabilities of the ontology.

Another possible topic of discussion is the need for some clear *quality criteria* of the CQs. For instance, all our CQs are universally closed formulas of the form

$$(P_1 \wedge P_2 \wedge \ldots \wedge P_n) \rightarrow P \qquad\qquad (21)$$

where P is not deducible from $\{P_1, P_2, \ldots, P_n\}$ without the help of the knowledge contained in the ontology. That is, none of our CQs includes information in the premises that should be inferred from the ontology. Note that this is not the case of some of the tests from the CSR problem domain of TPTP [30]. Thus, our set of CQs includes a *clean* version of these goals all the properties P_i in (21) that should be deducible from the ontology have been removed.

Finally, our set of CQs do not only have conjectures that are expected to be deducible, but it also contains conjectures expected not to be deducible. Obviously, this is also a very interesting experimentation, since these conjectures allow the detection of modelling errors when proved.

8. FUTURE WORK

The new framework presented in this paper opens multiple avenues for future research. Now, we are working in order to improve the competency of Adimen-SUMO. In the case of non-passing automatic tests, this implies to correct either a) the ontology itself, b) some mappings from WN to the ontology, or c) some WN relations. In parallel, we also want to enlarge our current set of CQs by gathering more questions from WN and its mapping to SUMO or alternative datasets. Additionally, it would be very interesting to determine which parts of the ontology are used to solve a set of CQs. Thus, in order to further debug Adimen-SUMO, we are also exploring the possibility to automatically derive an exhaustive set of questions from its axiomatization.

---

[10]A formal declarative semantics for the FO sublanguage SKIF of KIF is given in [12].

Our framework does not only allow to measure the competency of different SUMO-based ontologies, but also its efficiency when solving a large set of non-trivial inferences. In this sense, we are investigating more efficient representations of the ontology. Additionally, our framework can act as a new benchmark for testing the performance of FOL ATPs.

Another open research line will focus on proving the consistency of Adimen-SUMO. For instance, on the basis of a modular approach [19] or by including into our framework sophisticated model finders of the type of iProver [17].

Finally, we plan to develop automatic procedures to exploit Adimen-SUMO and its complete mapping to WN for automatically inferring new semantic properties and relations between WN concepts, or validating the consistency of resources associated to WN such as Cyc, DBpedia or Yago.

10. REFERENCES

[1] J. Álvez, P. Lucio, and G. Rigau. Adimen-SUMO: Reengineering an ontology for first-order reasoning. *Int. J. Semantic Web Inf. Syst.*, 8(4):80–116, 2012.

[2] G. Antoniou and F. v. Harmelen. *A Semantic Web Primer (2nd edition)*. MIT Press, 2008.

[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a web of open data. In Aberer, K. et al., editor, *The Semantic Web*, LNCS 4825, pages 722–735. Springer, 2007.

[4] C. Benzmüller and M. Ziener. Automated consistency checking of expressive ontologies - Beware of the wrong interpretation of success! In Fink, M. et al., editor, *Proc. of the 5th Int. Workshop on Acquisition, Representation and Reasoning with Contextualized Knowledge (ARCOE 2013)*, 2013.

[5] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - The story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[6] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia - A crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165, 2009.

[7] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[8] M. Fernández-López, A. Gómez-Pérez, and M. C. Suárez-Figueroa. Methodological guidelines for reusing general ontologies. *Data & Knowledge Engineering*, 86:242–275, 2013.

[9] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with DOLCE. In Gómez-Pérez A. et al., editor, *Knowledge Engin. and Knowledge Manag.: Ontologies and the Semantic Web*, LNCS 2473, pages 166–181. Springer, 2002.

[10] M. R. Genesereth, R. E. Fikes, D. Brobow, R. Brachman, T. Gruber, P. Hayes, R. Letsinger,

V. Lifschitz, R. Macgregor, J. Mccarthy, P. Norvig, R. Patil, and L. Schubert. Knowledge Interchange Format version 3.0 reference manual. Technical Report Logic-92-1, Stanford University, Computer Science Department, Logic Group, 1992.

[11] M. Grüninger and M. S. Fox. Methodology for the design and evaluation of ontologies. In *Proc. of the Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI 1995)*, 1995.

[12] P. Hayes and C. Menzel. A semantics for the Knowledge Interchange Format. In *Proc. of the Workshop on the IEEE Standard Upper Ontology (IJCAI 2001)*, 2001.

[13] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61, 2013.

[14] I. Horrocks and P. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(4):345–357, 2004.

[15] I. Horrocks and A. Voronkov. Reasoning support for expressive ontology languages using a theorem prover. In Dix J. et al., editor, *Foundations of Information and Knowledge Systems*, LNCS 3861, pages 201–218. Springer, 2006.

[16] P. Jain, P. Hitzler, P. Z. Yeh, K. Verma, and A. P. Sheth. Linked Data is merely more data. In Brickley D. et al., editor, *Proc. of the Spring Symposium: Linked Data Meets Artificial Intelligence*, pages 82–86. AAAI Press, 2010.

[17] K. Korovin. iProver - An instantiation-based theorem prover for first-order logic (system description). In Armando A. et al., editor, *Automated Reasoning*, LNCS 5195, pages 292–298. Springer, 2008.

[18] L. Kovács and A. Voronkov. First-order theorem proving and Vampire. In N. Sharygina and H. Veith, editors, *Computer Aided Verification*, LNCS 8044, pages 1–35. Springer, 2013.

[19] O. . Kutz and T. Mossakowski. A modular consistency proof for DOLCE. In Burgard W. et al., editor, *Proc. of the 25th AAAI Conf. on Artif. Intell. (AAAI 2011)*. AAAI Press, 2011.

[20] C. Matuszek, J. Cabral, M. J. Witbrock, and J. DeOliveira. An introduction to the syntax and content of Cyc. In C. Baral, editor, *Proc. of the Spring Symposium: Formalizing and Compiling Background Knowledge and Its Appl. to Knowledge Repr. and Question Answering*, pages 44–49. AAAI Press, 2006.

[21] I. Niles and A. Pease. Towards a standard upper ontology. In Guarino N. et al., editor, *Proc. of the 2nd Int. Conf. on Formal Ontology in Information Systems (FOIS 2001)*, pages 2–9. ACM, 2001.

[22] I. Niles and A. Pease. Linking lexicons and ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. In H. R. Arabnia, editor, *Proc. of the IEEE Int. Conf. on Inf. and Knowledge Engin. (IKE 2003)*, volume 2, pages 412–416. CSREA Press, 2003.

[23] A. Pease. Standard Upper Ontology Knowledge Interchange Format. Retrieved June 18, 2009, from `http://sigmakee.cvs.sourceforge.net/sigmakee/sigma/suo-kif.pdf`, 2009.

[24] A. Pease and G. Sutcliffe. First-order reasoning on a large ontology. In Sutcliffe G. et al., editor, *Proc. of the Workshop on Empirically Successful Automated Reasoning in Large Theories (CADE-21)*, CEUR Workshop Proceedings 257. CEUR-WS.org, 2007.

[25] R. Porzel and R. Malaka. A task-based approach for ontology evaluation. In *Proc. of the Workshop on Ontology Learning and Population (ECAI 2004)*, 2004.

[26] D. Ramachandran, R. P. Reagan, and K. Goolsbey. First-orderized ResearchCyc: Expressivity and efficiency in a common-sense ontology. In Shvaiko P. et al., editor, *Papers from the Workshop on Contexts and Ontologies: Theory, Practice and Applications (AAAI 2005)*, pages 33–40. AAAI Press, 2005.

[27] S. L. Reed and D. B. Lenat. Mapping ontologies into Cyc. In A. Pease, editor, *Papers from Workshop on Ontologies For The Semantic Web (AAAI 2002)*, pages 1–6. AAAI Press, 2002.

[28] A. Riazanov and A. Voronkov. The design and implementation of Vampire. *AI Communications*, 15(2-3):91–110, 2002.

[29] S. Schulz. E - A brainiac theorem prover. *AI Communications*, 15(2-3):111–126, 2002.

[30] G. Sutcliffe. The TPTP problem library and associated infrastructure. *J. Automated Reasoning*, 43(4):337–362, 2009.