Proceedings of the First International Workshop on Debugging Ontologies and Ontology Mappings - WoDOOM12

Galway, Ireland
October 8, 2012.

Edited by:
Patrick Lambrix
Guilin Qi
Matthew Horridge

Preface

Developing ontologies is not an easy task and, as the ontologies grow in size, they are likely to show a number of defects. Such ontologies, although often useful, also lead to problems when used in semantically-enabled applications. Wrong conclusions may be derived or valid conclusions may be missed. Defects in ontologies can take different forms. Syntactic defects are usually easy to find and to resolve. Defects regarding style include such things as unintended redundancy. More interesting and severe defects are the modeling defects which require domain knowledge to detect and resolve such as defects in the structure, and semantic defects such as unsatisfiable concepts and inconsistent ontologies. Further, during the recent years more and more mappings between ontologies with overlapping information have been generated, e.g. using ontology alignment systems, thereby connecting the ontologies in ontology networks. This has led to a new opportunity to deal with defects as the mappings and other ontologies in the network may be used in the debugging of a particular ontology in the network. It also has introduced a new difficulty as the mappings may not always be correct and need to be debugged themselves.

To deal with these issues a new workshop was created. This volume contains the proceedings of its first edition: WoDOOM12 - 1st International Workshop on Debugging Ontologies and Ontology Mappings held on October 8, 2012 in Galway, Ireland.

In his excellent invited talk, Bijan Parsia gave a classification of different defects in ontologies and discussed how easy or difficult it is to detect these defects. Further, there were presentations of two research papers and one experience paper, as well as a demonstration.

The editors would like to thank the Program Committee for their work in enabling the timely selection of papers for inclusion in the proceedings. We also appreciate our cooperation with EasyChair as well as our publisher Linköping University Electronic Press. WoDOOM12 was an EKAW 2012 (18th International Conference on Knowledge Engineering and Knowledge Management) workshop.

October 2012

Patrick Lambrix
Guilin Qi
Matthew Horridge

Workshop Organization

Workshop Organizers

Patrick Lambrix	Linköping University, Sweden
Guilin Qi	Southeast University, China
Matthew Horridge	Stanford University, USA

Program Committee

Oscar Corcho	Universidad Politécnica de Madrid, Spain
Bernardo Cuenca Grau	University of Oxford, UK
Jianfeng Du	Guangdong University of Foreign Studies, China
Peter Haase	fluid Operations, Germany
Aidan Hogan	Digital Enterprise Research Institute, Ireland
Matthew Horridge	Stanford University, USA
Ian Horrocks	University of Oxford, UK
Patrick Lambrix	Linköping University, Sweden
Yue Ma	Université Paris 13, France
Christian Meilicke	University of Mannheim, Germany
Nadeschda Nikitina	University of Karlsruhe, Germany
Bijan Parsia	University of Manchester, UK
Rafael Peñaloza	TU Dresden, Germany
Guilin Qi	Southeast University, China
Ulrike Sattler	University of Manchester, UK
Stefan Schlobach	Vrije Universiteit Amsterdam, The Netherlands
Bariş Sertkaya	SAP Research Dresden, Germany
Kostyantyn Shchekotykhin	University Klagenfurt
Kewen Wang	Griffith University, Australia
Peng Wang	Southeast University, China
Renata Wassermann	University of Sao Paulo, Brazil
Fang Wei-Kleiner	Linköping University, Sweden

Table of Contents

Measuring the Understandability of Deduction Rules for OWL

Tu Anh T. Nguyen, Richard Power, Paul Piwek, Sandra Williams

Department of Computing, The Open University, UK
{t.nguyen,r.power,p.piwek,s.h.williams}@open.ac.uk

Abstract. Debugging OWL ontologies can be aided with automated reasoners that generate entailments, including undesirable ones. This information is, however, only useful if developers understand *why* the entailments hold. To support domain experts (with limited knowledge of OWL), we are developing a system that explains, in English, why an entailment follows from an ontology. In planning such explanations, our system starts from a justification of the entailment and constructs a proof tree including intermediate statements that link the justification to the entailment. Proof trees are constructed from a set of intuitively plausible deduction rules. We here report on a study in which we collected empirical frequency data on the understandability of the deduction rules, resulting in a *facility index* for each rule. This measure forms the basis for making a principled choice among alternative explanations, and identifying steps in the explanation that are likely to require extra elucidation.

Keywords: Explanations, Entailments, Justifications, Understandability, Difficulty, Deduction Rules, Inference Rules

1 Introduction

An important tool in debugging ontologies is to inspect the entailments generated by automated reasoners such as FaCT++ [12] and Pellet [11]. An obviously incorrect entailed statement such as *SubClassOf(Person,Movie)* (Every person is a movie) signals that something has gone wrong, but many developers, especially those with limited knowledge of OWL, will need more information in order to make the necessary corrections: they need to understand *why* this undesirable entailment follows from the ontology, before they can start to repair it. A *justification* of an entailment—defined as any minimal subset of the ontology from which the entailment can be drawn [7]—provides a set of premises from which the entailment follows; however, user studies have shown that in many cases even OWL experts are unable to work out how the conclusion follows from the premises without further explanations [6]. Moreover, the opacity of standard OWL formalisms, which are designed for efficient processing by computer programs and not for fast comprehension by people, can be another obstacle for domain experts. As a possible solution to this problem, we are developing a system that explains, in English, why an entailment follows from an ontology.

To generate such explanations, our system starts from a justification of the entailment, which can be computed using the method described by Kalyanpur et al. [8], and constructs *proof trees* in which the root node is the entailment, the terminal nodes are the axioms in the justification, and other nodes are intermediate statements (i.e., lemmas). Proof trees are constructed from a set of intuitively plausible *deduction rules* which account for a large collection of deduction patterns, with each lemma introduced by a rule (as described in detail in [10]). For a given justification, the deduction rules might allow several proof trees, in which case we need a criterion for choosing the best.[1] From the selected proof tree, the system generates an English explanation. Such an explanation should be easier to understand than one based on the justification alone, as it replaces a single complex inference step with a number of simpler steps.

As an example, Table 1 shows an explanation generated by our prototype for the (obviously absurd) entailment "Every person is a movie", and based on the proof tree shown in Figure 1. The key to understanding this proof lies in the step from axiom 1 to statement (c), which is an example of an inference in need of "further elucidation"—a feature not yet implemented in our prototype.[2]

Table 1. An example explanation generated by our prototype

Input	**Entailment:** *SubClassOf(Person,Movie)* **Justification:** 1. *EquivalentClasses(GoodMovie,ObjectAllValuesFrom(hasRating,FourStars))* 2. *ObjectPropertyDomain(hasRating,Movie)* 3. *SubClassOf(GoodMovie,StarRatedMovie)* 4. *SubClassOf(StarRatedMovie,Movie)*
Output	Every person is a movie because the ontology implies that everything is a movie. Everything is a movie because (a) everything that has a rating is a movie, and (b) everything that has no rating at all is a movie. Statement (a) is from axiom 2 in the justification. Statement (b) is implied because (c) every--thing that has no rating at all is a good movie, and (d) every good movie is a movie. Statement (c) is implied because axiom 1 in the justification states that "a good movie is any--thing that has as rating only four stars". Statement (d) is implied because (e) every good movie is a star rated movie, and (f) every star rated movie is a movie. Statements (e) and (f) are from axioms 3 and 4 in the justification.

It is important to note that there may be multiple justifications for a given entailment in an ontology, and also multiple proof trees for a given justification. For either or both of these reasons, there may be multiple potential explanations for a given entailment, some of which may be easier to follow than others. Therefore, being able to identify the most understandable proof among alternatives would be of great help in planning an effective explanation.

[1] Alternatively the deduction rules might not yield any proof trees, in which case the system has to fall back on simply verbalising the justification. Obviously such cases will become rarer as we expand the set of rules.

[2] Axiom 1 asserts an equivalence between two classes: good movies, and things that only have ratings of four stars. The precise condition for an individual to belong to the second class is that all of its ratings should be four star, and this condition would be trivially satisfied *if the individual had no ratings at all.*

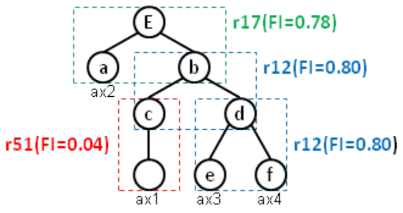


Fig. 1. The proof tree of the explanation in Table 1. The labels r17 etc. refer to rules listed in Table 3. FI values represent how easy it is to understand the rules—their *Facility Indexes*—with values ranging from 0.0 (hardest) to 1.0 (easiest).

This paper focusses on the deduction rules and their understandability. We describe how our current set of deduction rules was collected through analysis of a large corpus of approximately 500 OWL ontologies, and report on an empirical study that allows us to assign easiness levels to the deduction rules.[3] This facility index provides a basis for measuring the understandability of an entire explanation and for making a principled choice among alternative explanations. It also indicates which steps in an explanation are likely to be difficult and in need of extra elucidation—for example, the inference from axiom 1 to statement (c) in the explanation in Table 1. We envisage that the method described here can be used by others to empirically test different sets of deduction rules.

2 Deduction Rules

Intuitively, a deduction rule is an inferential step from premises to a conclusion, which cannot be effectively simplified by introducing substeps (and hence, intermediate conclusions). In practice this means that deduction rules have relatively few premises, and in fact we limit this number to four. Formally speaking, both the conclusion and the premises are OWL axioms, but they are generalised by using variables that abstract over class and property names. An example of our deduction rules is $SubClassOf(X,Y) \wedge SubClassOf(Y,Z) \rightarrow SubClassOf(X,Z)$ (rule 12), which corresponds to the well-know syllogism that from “Every X is a Y” and “Every Y is a Z” we may infer “Every X is a Z”.

Our deduction rules were derived empirically through a corpus study of around 500 OWL ontologies. These were collected from a variety of sources, including the TONES repository [2], the Swoogle search engine [3] and the Ontology Design Patterns corpus [1]; they thus cover a wide range of topics and authoring styles. To collect deduction rules, we first computed entailment-justification pairs using the method described by Nguyen et al. [9], then collated them to obtain a list of deduction patterns ranked by frequency. From this list,

[3] In a deduction rule, the premises can be viewed as a justification of the entailment. Horridge et al. proposed a model for measuring the difficulty of a justification [4], but this model was based on the complexity of its logical structure of the justification rather than its difficulty for people.

we selected deduction patterns that were simple (in a sense that will be explained shortly) and frequent, such as rule r12 mentioned above. Subsequently we added some further rules that occurred often as *parts* of more complex deduction patterns, but were not computed as separate patterns because of certain limitations of the reasoning algorithms.[4] An example of such rules is $\mathit{ObjPropDom(r0,X)} \wedge \mathit{SubClassOf(ObjectAllValuesFrom(r0,\bot),X)} \rightarrow \mathit{SubClassOf(\top,X)}$ (rule 17), which is from "Everything that r0 something is an X" and "Everything that r0 nothing at all is an X" we infer "Everything is an X".

As a criterion of *simplicity* we considered the number of premises (we stipulated not more than four) and also what is called the *laconic* property [5]—that an axiom should not contain information that is not required for the entailment to hold. We have assumed that deduction rules that are simple in this sense are more likely to be more *understandable* by most people. So far, 57 deduction rules have been obtained in this way. These rules are sufficient to generate appropriate lemmas for 48% of the justifications of subsumption entailments in the corpus (i.e., over 30,000 justifications).

3 Measuring Understandability

3.1 Materials

To measure the understandability of a rule, we devised a deduction problem in which premises of the rule were given in English, replacing class or property variables by fictional nouns and verbs so that the reader would not be biassed by domain knowledge, and the subjects were asked whether the entailment of the rule followed from the premises. The correct answer was always "Follows", so to control for positive response bias (i.e., favouring a positive answer to *any* question) we included questions for non-entailments and trivial entailments, which we will call *control questions* as opposed to *test questions.*

Our control questions were designed to be obvious to subjects who did the test seriously (rather than responding casually without reading the problem properly). Specifically, they either repeated one of the premises (in which case, trivially, the correct answer was "Follows"), or made statements about objects not mentioned in the premises (in which case, also trivially, the correct answer was "Does not Follow"). Problems consisted of premises followed by two questions, one a test question and one a control; for half the problems the correct answers were "Follows" and "Follows", for the other half "Follows" and "Does not Follow", with question order varied so that the test question sometimes preceded the control, and sometimes followed it.

3.2 Method

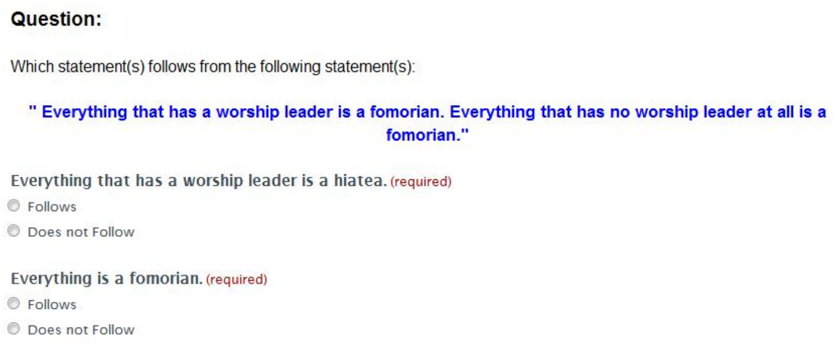
The study was conducted on CrowdFlower, a crowdsourcing service that allows customers to upload tasks to be passed to labour channel partners such as Ama-

[4] Reasoning services for OWL typically compute only some kinds of entailment, such as subclass and class membership statements, and ignore others.

zon Mechanical Turk.[5] We set up the operation so that tasks were channelled only to Amazon's Mechanical Turk, and were restricted to subjects from Australia, the United Kingdom and the United States since we were aiming to recruit as many (self-reported) native speakers of English as possible.

To eliminate responses from 'scammers' (people who respond casually without considering the problem seriously), we used CrowdFlower's quality control service which is based on *gold-standard data*: customers provide problems called *gold units* for which the correct answer is specified, allowing CrowdFlower to filter automatically any subjects whose performance on gold units falls below a threshold (75%). Our gold units resembled our test units, each having premises followed by two questions, but both questions were control units with answers that should have been obvious to any serious subject. The management of gold units is internal to CrowdFlower, so these data are not included in our analysis.

Of the 57 deduction rules we collected, 51 rules were measured in this way. For example, rule r17 (from Figure 1) was measured based on data gathered from the problem in Figure 2, with the rule's entailment as the second question. It is important to note that in CrowdFlower subjects are not required to complete all problems. They can give up whenever they want, and their responses will be accepted so long as they perform well on gold units. CrowdFlower randomly assigns non-gold problems to subjects until it collects up to a specified number of valid responses for each problem; in our study we specified 50, but since some subjects gave up part-way through, the number of subjects was over 100.


Question:

Which statement(s) follows from the following statement(s):

" Everything that has a worship leader is a fomorian. Everything that has no worship leader at all is a fomorian."

Everything that has a worship leader is a hiatea. (required)
- Follows
- Does not Follow

Everything is a fomorian. (required)
- Follows
- Does not Follow

Fig. 2. The testing problem for rule r17—*ObjPropDom(r0,X)* $\wedge$ *SubClassOf(ObjectAllValuesFrom(r0,$\perp$),X)* $\rightarrow$ *SubClassOf($\top$,X)*

4 Results

The main aim of the study was to collect frequency data on whether people recognise that the conclusion of a (verbalised) deduction rule follows from the

[5] See `http://crowdflower.com/` and `http://www.mturk.com/` for details.

premises. However, these data provide a valid index of understandability only if we can control for positive response bias: in the extreme case, a subject that always gives the positive answer ("Follows" rather than "Does not follow") will get all the test questions right, regardless of their difficulty. We used control questions to address this issue—additional to the CrowdFlower gold-unit filtering. The use of control questions in each problem also provided an opportunity for confirming that in general subjects were taking the survey seriously.

4.1 Control Questions

Figure 3 shows that for the 108 subjects that participated in the study, all answered around 75% or more of the control questions correctly, suggesting that they were performing the task seriously.

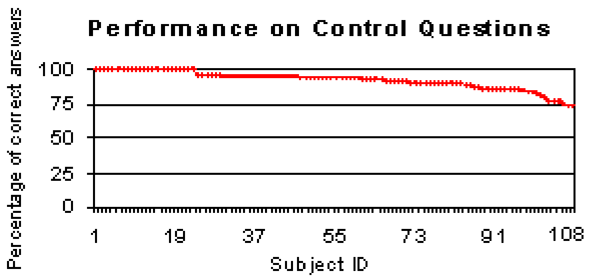


Fig. 3. The subjects' performance on the control questions

4.2 Response Bias

Table 2 shows the absolute frequencies of the responses "Follows" (+F) and "Does not follow" (−F) for all non-gold questions in the study—control as well as test. It also subdivides these frequencies according to whether the answer was correct (+C) or incorrect (−C). Thus for example the cell +F+C counts cases in which subjects answered "Follows" when this was the correct answer, while +F−C counts cases in which they answered "Follows" when this was incorrect.

Recalling that for half the problems the correct answers were +F+F, while for half they were +F−F, the percentage of +F answers for a subject that always answered correctly would be 75%. If subjects had a positive response bias we would expect an overall rate higher than this, but in fact we obtain 3617/4930 or 73.4%, suggesting little or no bias in either direction.

Looking at the distribution of incorrect answers, we can also ask whether subjects erred through being too ready to accept invalid conclusions (−F−C), or too unwilling to accept conclusions that were in reality valid (+F−C). The table shows a clear tendency towards the latter, with only 118 responses in

−F−C compared with an expected value of 274 (1030*1313/4930) calculated from the overall frequencies. In other words, subjects were more likely to err by rejecting a valid conclusion than by accepting an invalid one, a finding confirmed statistically by the highly significant association between response (±F) and correctness (±C) on a 2×2 chi-squared test ($\chi^2 = 153.5$, df = 1, $p < 0.0001$).

Table 2. The distribution of the subjects' performance

	+F	**-F**	**TOTAL**
+C	2705	1195	3900
-C	912	118	1030
TOTAL	3617	1313	4930

4.3 Facility Indexes

We use the proportion of correct answers for each test question as an index of understandability of the associated deduction rule, which we will call its *facility index*. This index provides our best estimate of the probability that a person will understand the relevant inference step—i.e., that they will recognise that the conclusion follows from the premise—and accordingly ranges from 0.0 to 1.0. Values of the facility index for the rules tested in the study are shown in Table 3, ordered from high values to low. In this table, the rules r12 and r17 used in the explanation in Table 1 are relatively easy, with facility indexes of 0.80 and 0.78. By contrast rule r51, which infers statement (c) from axiom 1 in the example, is the hardest, with a facility index of only 0.04, and hence evidently a step in need of further elucidation—for instance as follows:

> Statement (c) is inferred from axiom 1, which asserts an equivalence between two classes: 'good movie' and 'anything that has as rating only four stars'. Since the second class trivially accepts anything that has no rating at all, we conclude that anything that has no rating at all is a good movie.

It can be seen in the table that for closely related rules, such as r11, r12 and r14, the facility indexes are quite close to each other (see also r17 and r19), a result that confirms the reliability of the values.

5 Conclusion

The main aim of this paper is to report empirical results on the difficulty of some inference steps that often occur in proofs, in particular for entailments computed from ontologies. These results allow us to estimate the understandability of proofs that can serve as the basis for verbal explanations of entailments, so making it clear to an ontology developer why an undesired statement was inferred, and which axiom(s) in the ontology should be removed or revised.

In our explanation planner, the facility indexes for the deduction rules are used in two ways. First, by combining the values for all the rules in a given proof tree, we can estimate the difficulty of the whole tree, and thus make a principled choice among alternative trees. If we think of facility indexes as measuring the probability that a reader will understand a given step in the explanation, a natural method of combining indexes would be to multiply them, so computing the joint probability of all steps being followed; the higher this value, obviously, the better. Second, once a proof tree has been chosen as more understandable than the alternatives (if any), we can apply the indexes again by looking for steps that are relatively hard, and considering whether to add extra elucidation at that point. We plan to do this by investigating a range of explanation strategies for each difficult rule, and determining empirically which is most effective.

Leaving aside the way facility indexes are used in our work, we believe both the indexes and our method for obtaining them are worth for reporting as a resource for other researchers, who might find them useful in alternative models or contexts.

References

1. Ontology Design Patterns. `http://ontologydesignpatterns.org`, Last Accessed: 30th August 2010
2. The TONES Ontology Repository. `http://owl.cs.manchester.ac.uk/repository/`, Last Accessed: 30th August 2010
3. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: ACM International Conference on Information and Knowledge Management (CIKM 2004). pp. 652–659 (2004)
4. Horridge, M., Bail, S., Parsia, B., Sattler, U.: The Cognitive Complexity of OWL Justifications. In: International Semantic Web Conference (ISWC 2011). pp. 241–256 (2011)
5. Horridge, M., Parsia, B., Sattler, U.: Laconic and Precise Justifications in OWL. In: International Semantic Web Conference (ISWC 2008). pp. 323–338 (2008)
6. Horridge, M., Parsia, B., Sattler, U.: Lemmas for Justifications in OWL. In: International Workshop on Description Logics (DL 2009) (2009)
7. Kalyanpur, A.: Debugging and repair of OWL ontologies. Ph.D. thesis, The University of Maryland, US (2006)
8. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding All Justifications of OWL DL Entailments. In: International Semantic Web Conference (ISWC 2007) (2007)
9. Nguyen, T.A.T., Piwek, P., Power, R., Williams, S.: Justification Patterns for OWL DL Ontologies. Tech. Rep. TR2011/05, The Open University, UK (2010)
10. Nguyen, T.A.T., Power, R., Piwek, P., Williams, S.: Planning Accessible Explanations for Entailments in OWL Ontologies. In: International Natural Language Generation Conference (INLG 2012). pp. 110–114 (2012)
11. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A Practical OWL-DL Reasoner. Journal of Web Semantics 5, 51–53 (2007)
12. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic Reasoner: System Description. In: International Joint Conference on Automated Reasoning (IJCAR 2006). pp. 292–297 (2006)

Table 3: Deduction rules and their facility indexes (FI), with ‘CA’ means the absolute number of correct answers and ‘S’ means the absolute number of subjects. For short, the names of OWL functors are abbreviated.

ID	Rule	Deduction Problem	CA	S	FI
1	EqvCla(X,Y...) →SubClaOf(X,Y)	A hiatea is defined as a milvorn. →Every hiatea is a milvorn.	49	49	1.00
2	SubClaOf(X,ObjIntOf(Y,Z...)) →SubClaOf(X,Y)	Every ormyrr is both a gargoyle and a harpy. →Every ormyrr is a gargoyle.	47	49	0.96
3	ObjPropDom(r0,X) ∧ SubClaOf(X,Y) →ObjPropDom(r0,Y)	Anything that has a supernatural ability is a bulette. Every bulette is a manticore. →Anything that has a supernatural ability is a manticore.	45	47	0.96
4	SubClaOf(ObjUniOf(Y,Z...),X) →SubClaOf(Y,X)	Everything that is a volodni or a treant is a maradan. →Every volodni is a maradan.	44	46	0.96
5	SubClaOf(X,Y) ∧ SubClaOf(X,Z) →SubClaOf(X,D⊓Z)	Every bullywug is a grippli. Every bullywug is a prismatic. →Every bullywug is both a grippli and a prismatic.	45	48	0.94
6	SubClaOf(⊤,X) →SubClaOf(Y,X)	Everything is a kelpie. →Every person is a kelpie.	43	46	0.93
7	SubClaOf(X,ObjSomValF(r0,⊤)) ∧ SubClaOf(X,ObjAllValF(r0,Y)) →SubClaOf(X,ObjSomValF(r0,Y))	Every locathah eats something. Every locathah eats only orogs. →Every locathah eats an orog.	45	50	0.90
8	ObjPropRng(r0,Y) ∧ SubClaOf(Y,X) →ObjPropRng(r0,X)	Anything that something lives in is a tarrasque. Every tarrasque is a kraken. →Anything that something lives in is a kraken.	44	49	0.90
9	ObjPropDom(r0,X) ∧ SubClaOf(Y,ObjSomValF(r0,Z)) →SubClaOf(Y,X)	Anything that is a messenger of something is a landwyrm. Every spellgaunt is a messenger of a gravorg. →Every spellgaunt is a landwyrm.	43	50	0.86
10	EqvCla(X,ObjUniOf(Y,Z...)) →SubClaOf(Y,X)	Every cooshee is a peryton or a banderlog; everything that is a peryton or a banderlog is a cooshee. →Every peryton is a cooshee.	41	50	0.82
11	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(ObjMinCard(1,r0,Y),Z)) →SubClaOf(X,Z)	Every varag lives on a seaplane. Everything that lives on at least one seaplane is an urophion. →Every varag is an urophion.	40	49	0.82
12	SubClaOf(X,Y) ∧ SubClaOf(Y,Z) →SubClaOf(X,Z)	Every sivak is a draconian. Every draconian is a guulvorg. →Every sivak is a guulvorg.	37	46	0.80
13	SubClaOf(X,ObjCompOf(X)) →SubClaOf(X,⊥)	Every zezir is something that is not a zezir. →Nothing is a zezir.	40	50	0.80
14	SubObjPpOf(r0,r1) ∧ SubObjPpOf(r1,r2) →SubObjPpOf(r0,r2)	The property ”is a kobold of” is a sub-property of ”is a drow of”. The property ”is a drow of” is a sub-property of ”is a tiefling of”. →The property ”is a kobold of” is a sub-property of ”is a tiefling of”.	41	52	0.79

Continued on Next Page...

15	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(Y,Z) →SubClaOf(X,ObjSomValF(r0,Z))	Every phaerlin is father of a firbolg. Every firbolg is a gnoll. →Every phaerlin is father of a gnoll.	37	47	0.79
16	EqvCla(X,ObjIntOf(Y,Z. . .)) →SubClaOf(X,Y)	A cyclops is anything that is both a troofer and a gathra. →Every cyclops is a troofer.	37	47	0.79
17	ObjPropDom(r0,X) ∧ SubClaOf(ObjAllValF(r0,⊥),X) →SubClaOf(⊤,X)	Everything that has a worship leader is a fomorian. Everything that has no worship leader at all is a fomorian. →Everything is a fomorian.	39	50	0.78
18	ObjPropRng(r0,X) ∧ SymObjProp(r0) →ObjPropDom(r0,X)	Anything that something is an abrian of is a grolantor. X is an abrian of Y if and only if Y is an abrian of X. →Anything that is a sibling of something is a grolantor.	36	47	0.77
19	SubClaOf(Y,X) ∧ SubClaOf(ObjCompOf(Y),X) →SubClaOf(⊤,X)	Every oblivion moss is a vegepygmy. Everything that is not an oblivion moss is a vegepygmy. →Everything is a vegepygmy.	36	47	0.77
20	ObjPropDom(r0,⊥) →SubClaOf(⊤,ObjAllValF(r0,⊥))	There does not exist anything that is a grimlock of something. →Everything is not a grimlock.	39	51	0.76
21	ObjPropRng(r0,⊥) →SubClaOf(⊤,ObjAllValF(r0,⊥))	There does not exist anything that something has as a catter. →Everything has no catter at all.	37	49	0.76
22	DisCla(X,Y. . .) ∧ SubClaOf(Z,X) ∧ SubClaOf(W,Y) →DisCla(Z,W)	No plant is an animal. Every kalamanthis is a plant. Every tendriculos is an animal. →No kalamanthis is a tendriculos.	35	46	0.76
23	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(Y,ObjSomValF(r0,Z)) ∧ TrnObjProp(r0) →SubClaOf(X,ObjSomValF(r0,Z))	Every dero is a tendriculos of a harpy. Every harpy is a tendriculos of a tasloi. If X is a tendriculos of Y and Y is a tendriculos of Z then X is a tendriculos of Z. →Every dero is a tendriculos of a tasloi.	38	51	0.75
24	SubClaOf(X,ObjUniOf(Y,Z)) ∧ SubClaOf(Y,W) ∧ SubClaOf(Z,W) →SubClaOf(X,W)	Every mongrelfolk is a nilbog or a norker. Every nilbog is a skulk. Every norker is a skulk. →Every mongrelfolk is a skulk.	35	48	0.73
25	SubClaOf(ObjCompOf(X),Y) →SubClaOf(⊤,C⊔Y)	Everything that is not a spriggan is an orog. →Everything is a spriggan or an orog.	36	50	0.72
26	SubClaOf(X,ObjUniOf(Y,Z)) ∧ SubClaOf(Y,Z) →SubClaOf(X,Z)	Every merfolk is a lizardfolk or a kobold. Every lizardfolk is a kobold. →Every merfolk is a kobold.	35	49	0.71
27	SubClaOf(ObjSomValF(r0,X),Y) ∧ SubClaOf(ObjAllValF(r0,⊥),Y) →SubClaOf(ObjAllValF(r0,X),Y)	Everything that supervises a worg is a stirge. Everything that supervises nothing at all is a stirge. →Everything that supervises only worgs is a stirge.	35	49	0.71
28	ObjPropDom(r0,X) ∧ SymObjProp(r0)	Anything that is an obliviax of something is a kraken. X is an obliviax of Y if and only if Y is an obliviax of X.	34	49	0.69

Continued on Next Page. . .

	→ObjPropRng(r0,X)	→Anything that something is an obliviax of is a kraken.			
29	SubClaOf(X,ObjSomValF(r0,ObjSomValF(r0,Y))) ∧ TrnObjProp(r0) →SubClaOf(X,ObjSomValF(r0,Y))	Every draconian is a spriggan of something that is a spriggan of a shifter. If X is a spriggan of Y and Y is a spriggan of Z then X is a spriggan of Z. →Every draconian is a spriggan of a shifter.	34	50	0.68

30	ObjPropRng(r0,Z)
	∧ SubClaOf(X,ObjSomValF(r0,Y))
	→SubClaOf(X,ObjSomValF(r0,
	ObjIntOf(Y,Z)))
		Anything that something resembles is a corollax.
		Every mudmaw resembles a jermlaine.
		→Every mudmaw resembles something that is both
		a jermlaine and a corollax.
			32	50	0.64

31	SubClaOf(⊤,Y) ∧ DisCla(X,Y) →SubClaOf(X,⊥)	Everything is a darfellan. No grippli is a darfellan. →Nothing is a grippli.	30	47	0.64
32	SubClaOf(X,ObjExtCard(n1,r0,Y)) →SubClaOf(X,ObjMinCard(n2,r0,Y)) where n2 ≤ n1	Every oaken defender has exactly two dry leaves. →Every oaken defender has at least one dry leaf.	29	46	0.63
33	ObjPropDom(r0,X) ∧ SubObjPpOf(r1,r0) →ObjPropDom(r1,X)	Anything that gyres something is a tiefling. The property ”raths” is a sub-property of ”gyres”. →Anything that raths something is a tiefling.	28	46	0.61
34	SubClaOf(X,Y) ∧ DisCla(X,Y) →SubClaOf(X,⊥)	Every aasimar is a sirine. No aasimar is a sirine. →Nothing is an aasimar.	30	53	0.57
35	SubClaOf(X,Y) ∧ SubClaOf(X,Z) ∧ DisCla(Y,Z) →SubClaOf(X,⊥)	Every needleman is a basidirond. Every needleman is a battlebriar. No basidirond is a battlebriar. →Nothing is a needleman.	27	48	0.56
36	TrnObjProp(r0) ∧ InvObjProp(r0,r1) →TrnObjProp(r1)	If X toves Y and Y toves Z then X toves Z. X toves Y if and only if Y is toved by X. →If X is toved by Y and Y is toved by Z then X is toved by Z.	27	49	0.55
37	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubObjPpOf(r0,r1) →SubClaOf(X,ObjSomValF(r1,Y))	Every halfling is an ascomoid of a kenku. The property ”is an ascomoid of” is a sub-property of ”is a basidirond of”. →Every halfling is a basidirond of a kenku.	28	51	0.55
38	ObjPropRng(r1,X) ∧ SubObjPropOf(r0,r1) →ObjPropRng(r0,X)	Anything that something brilligs is a girallon. The property ”gimbles” is a sub-property of ”brilligs”. →Anything that something gimbles is a girallon.	24	46	0.52
39	SubClaOf(X,Y) ∧ SubClaOf(X,ObjCompOf(Y)) →SubClaOf(X,⊥)	Every darkmantle is a gorgon. Every darkmantle is not a gorgon. →Nothing is a darkmantle.	25	49	0.51
40	SubClaOf(X,ObjSomValF(r0,ObjIntOf(Y,Z. . .))) ∧ DisCla(Y,Z) →SubClaOf(X,⊥)	Every daemonfey is preceded by something that is both an axani and a phoera. No axani is a phoera. →Nothing is a daemonfey.	25	50	0.50
41	SubClaOf(X,ObjMinCard(n1,r0,Dor⊤)) ∧ SubClaOf(X,ObjMinCard(n2,r0,⊤)), 0 < n2 < n1	Every jermlaine possesses at least three things. Every jermlaine possesses at most one thing.	22	46	0.48

Continued on Next Page. . .

	→SubClaOf(X,⊥)	→Nothing is a jermlaine.			
42	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(Y,⊥) →SubClaOf(X,⊥)	Every tasloi has as owner an aasimar. Nothing is an aasimar. →Nothing is a tasloi.	20	44	0.45
43	FunDatProp(d0) ∧ SubClaOf(X,DataMinCard(n,d0,DR0)), n > 1 where n > 1 →SubClaOf(X,⊥)	Everything has as ratings at most one value. Every buckawn has as ratings at least four integer values. →Nothing is a buckawn.	20	49	0.41
44	ObjPropRng(r0,X) ∧ InvObjProp(r0,r1) →ObjPropDom(r1,X)	Anything that something gimbles from is a terlen. X gimbles from Y if and only if Y gimbles into X. →Anything that gimbles into something is a terlen.	19	47	0.40
45	FunDatProp(d0) ∧ SubClaOf(X,DataHasVal(d0,l0⋆DT0)) ∧ SubClaOf(X,DataHasVal(d0,l1⋆DT1)) where DT0 and DT1 are disjoint or l0 ≠ l1 →SubClaOf(X,⊥)	Everything has as power level at most one value. Every sirine has as power level an integer value of 5. Every sirine has as power level an integer value of 7. →Nothing is a sirine.	18	45	0.40
46	FuncObjProp(r0) ∧ SubClaOf(X,ObjHasVal(r0,i0)) ∧ SubClaOf(X,ObjHasVal(r0,i1)) ∧ DiffInd(i0,i1. . .) →SubClaOf(X,⊥)	Everything worships at most one thing. Every selkie worships Ashur. Every selkie worships Enki. Ashur and Enki are different individuals. →Nothing is a selkie.	17	44	0.39
47	ObjPropDom(r1,X) ∧ InvObjProp(r1,r0) →ObjPropRng(r0,X)	Anything that gimbles from something is an atomie. X gimbles from Y if and only if Y gimbles into X. →Anything that something gimbles into is an atomie.	18	48	0.38
48	SubClaOf(X,ObjAllValF(r0,Y) ∧ InvObjProp(r0,r1) →SubClaOf(ObjSomValF(r1,X),Y)	Every tabaxi toves from only lamias. X toves from Y if and only if Y toves into X. →Everything that toves into a tabaxi is a lamia.	16	50	0.32
49	DataPropRange(d0,DR0) ∧ SubClaOf(X,ObjSomValF(r0,DataHasVal(d0, l0⋆DT1))) where DR0 & DT1 are disjoint →SubClaOf(X,⊥)	Any value that something has as dark-vision is an integer value. Every ettin makes friends with something that has as dark-vision string value of ”three”. String values are unconvertible to integer values in OWL. →Nothing is an ettin.	9	48	0.19
50	DataPropRange(d0,DR0) ∧ SubClaOf(X,DataSomeValFrm(d0,DT1)) where DR0 & DT1 are disjoint →SubClaOf(X,⊥)	Any value that something has as life expectancy is an integer value. Every tiefling has as life expectancy a double value. Double values are unconvertible to integer values in OWL. →Nothing is a tiefling.	9	49	0.18
51	EqvCla(X,ObjAllValF(r0,Y) →SubClaOf(ObjAllValF(r0,⊥),X)	A hiatea is anything that eats only lamias. →Everything that eats nothing at all is a hiatea.	2	49	0.04

Declutter Your Justifications: Determining Similarity Between OWL Explanations

Samantha Bail, Bijan Parsia, Ulrike Sattler

The University of Manchester
Oxford Road, Manchester, M13 9PL
{bails,bparsia,sattler@cs.man.ac.uk}

Abstract. Given the high expressivity of the Web Ontology Language OWL 2, there is a potential for great diversity in the logical content of OWL ontologies. The fact that many naturally occurring entailments of such ontologies have multiple justifications indicates that ontologies often overdetermine their consequences, suggesting a diversity in supporting reasons. On closer inspection, however, we often find that justifications – even for multiple entailments – appear to be structurally similar, suggesting that their multiplicity might be due to diverse *material*, not *formal* grounds for an entailment.
In this paper, we introduce and explore several equivalence relations over justifications for entailments of OWL ontologies which partition a set of justifications into structurally similar subsets. These equivalence relations range from strict isomorphism to looser notions of similarity, covering justifications which contain different class expressions, or even different numbers of axioms. We present the results of a survey of 83 ontologies from the bio-medical domain, showing that OWL ontologies used in practice often contain large numbers of structurally similar justifications.

1 Introduction

Justifications, minimal entailing subsets of an OWL[1] ontology, provide helpful and easy-to-understand explanation support when repairing unwanted entailments in the ontology debugging process. They are currently the prevalent form of explanation in OWL ontology editors such as Protégé 4. While we have some knowledge of how individual justifications can be made easier to understand for human users, e.g. [8,11], we have yet to gain more insights into user interaction with *multiple justifications*. An entailment of an OWL ontology can have a large number of justifications (potentially exponential in the number of axioms in the ontology [3]), with up to several hundreds found in large real-life ontologies [4]. In order to achieve a *minimal repair*, i.e. a modification of the ontology which removes unwanted entailments without losing relevant information, it is often beneficial to consider not only a single, but multiple entailments simultaneously.

[1] We will use the term *OWL* to refer to both OWL and OWL 2 ontologies.

When encountering justifications for a finite set of entailments of an ontology (e.g. unwanted atomic subsumptions, or unsatisfiable classes), we are often faced with a seemingly large and diverse body of reasons why the entailments hold. *Root and derived* justifications [13,14] address this issue by pointing out those justifications which are subsets of others; fixing such a subset (*root*) justifications first will also repair those justifications which are *derived* from it. While proven to be helpful, root and derived justifications are restricted to a very specific kind of relation between justifications. Due to a lack of other suitable interaction strategies, large numbers of multiple justifications may still present themselves to a user as an unordered and often unmanageable list of axiom sets.

On closer inspection, however, we frequently find that sets of justifications are very similar, and often even contain structurally identical axioms, with only class, property, and individual names diverging. Pointing out these similarities and grouping justifications based on their shared structures might greatly assist a user in coping with multiple justifications: Rather than trying to understand each individual material justification, the user can focus on understanding the formal *template* of a particular subset of justifications. Potentially, a user might have to deal with far fewer justifications, thus having a significantly reduced effort when repairing an ontology. This raises two questions: First, how do we determine whether two justifications are structurally similar, and second, how prevalent are such similarities in ontologies used in practice?

A well-known syntactical equivalence relation in OWL is *structural equivalence*. The OWL Structural Specification[2] states the condition for two OWL objects (named classes, properties, or individuals, complex expressions, or OWL axioms) to be structurally equivalent. In short, it defines the objects to be equivalent if they contain the same complex expressions, using identical entity names and constructors, regardless of ordering and repetition (in an unordered association). The OWL API,[3] a Java API which is used to manipulate OWL ontologies, implements this notion of structural equivalence by default.

A looser notion of structural similarity, *justification isomorphism* [6], was first introduced in a study of the cognitive complexity of justifications: Two justifications are isomorphic if there exists a mapping between class, property and individual names of the justifications which makes them structurally equivalent. This equivalence relation covers justifications which contain the same number of axioms, constructors, as well as class, property, and individual names. Justification isomorphism has previously been shown to significantly reduce a corpus of justifications from 64,800 to merely 11,600 justification templates [6].

While justification isomorphism helps to eliminate the effects of diverging entity names, we can also identify types of justifications which may be considered to be very similar despite their use of different constructors:

[2] http://www.w3.org/TR/owl2-syntax
[3] http://owlapi.sourceforge.net

Example 1

$$\mathcal{J}_1 = \{A \sqsubseteq B \sqcap C, B \sqcap C \sqsubseteq D\} \models A \sqsubseteq D$$
$$\mathcal{J}_2 = \{A \sqsubseteq \exists r.C, \exists r.C \sqsubseteq D\} \models A \sqsubseteq D$$

In this example, the semantics of the complex expressions $B \sqcap C$ in $\mathcal{J}_1$ and $\exists r.C$ in $\mathcal{J}_2$ are not relevant for the respective entailment; their occurrences in the justifications and their entailments could be replaced by freshly generated atomic concept names without affecting the entailment relation. Such a substitution in turn would make the two justifications isomorphic.

Likewise, justifications of different lengths may be considered similar if their general structure of reasoning is identical:

Example 2

$$\mathcal{J}_1 = \{A \sqsubseteq B, B \sqsubseteq C\} \models A \sqsubseteq C$$
$$\mathcal{J}_2 = \{A \sqsubseteq B, B \sqsubseteq C, C \sqsubseteq D\} \models A \sqsubseteq D$$

These two justifications clearly require the same form of reasoning from a user, namely the understanding of simple atomic subsumption. Strict isomorphism only applies to justifications which contain the same number of axioms; it does not cover situations like the above. However, for the purpose of structuring sets of justifications and analysing the logical diversity of a corpus of justifications, capturing those kinds of similarities illustrated in the above examples would be highly desirable.

The idea of finding similarities between concepts in Description Logics has been widely explored in the work on *unification* and *matching*, e.g. [1,2], for the purpose of detecting redundant concept descriptions in knowledge bases. The aim of unification is to find a suitable substitution σ which maps atomic concepts in a concept term C to (possibly non-atomic) concepts in a concept term D such that the two terms are made equivalent.

While unification and matching are very close to our requirements for capturing similarities between justifications, the concepts are not directly applicable. In our case, the inputs are of different shape from the matching problem: The goal is to unify two sets of axioms and the corresponding entailments, rather than matching a given concept pattern containing variables with a concept description.

The above examples motivate a looser notion of justification isomorphism, which allows us to identify justifications as equivalent if they require the same reasoning mechanisms, regardless of size, signature, and logical constructors used. In the present paper, we introduce two new types of equivalence relations based on matching *subexpressions* and *lemmas*, and analyse the effect these extended relations have when applied to a corpus of justifications from the bio-medical domain.

2 Preliminaries

2.1 Justifications in OWL

We assume the reader to be familiar with OWL and the underlying Description Logic $\mathcal{SROIQ}$ [9]. In what follows, $A, B, \ldots$ denote class names in an ontology $\mathcal{O}$, r, s role names, and α denotes an OWL axiom.

The concept of pinpointing minimal entailing subsets of an ontology is the dominant form of explanation for entailments of OWL ontologies [15,13,3]. A justification (also denoted as *MinA*, or MUPS when referring to unsatisfiable classes) is defined as a minimal subset of an ontology $\mathcal{O}$ that causes an entailment η to hold:

Definition 1 (Justification) *$\mathcal{J}$ is a justification for $\mathcal{O} \models \eta$ if $\mathcal{J} \subseteq \mathcal{O}, \mathcal{J} \models \eta$ and, for all $\mathcal{J}' \subset \mathcal{J}$, it holds that $\mathcal{J}' \not\models \eta$.*

For every axiom which is asserted in the ontology, the axiom itself naturally is a justification. We call such a justification a *self-justification*, and an entailment which has only a self-justification and no other justification in $\mathcal{O}$ a *self-supporting* entailment.

It is important to note that a justification is always defined with respect to an entailment η. In the remainder of this paper we will therefore use the term *justification* to describe a justification-entailment pair $(\mathcal{J}, \eta)$ where $\mathcal{J}$ is a minimal entailing axiom set for η.

2.2 Justification Isomorphism

Isomorphism between justifications was first introduced as a method to reduce the number of similar justifications when sampling from a large corpus to justifications [6].

Definition 2 (Justification Isomorphism) *Two justifications $(\mathcal{J}_1, \eta_1)$, $(\mathcal{J}_2, \eta_2)$ are* isomorphic *($(\mathcal{J}_1, \eta_1) \approx_i (\mathcal{J}_2, \eta_2)$) if there exists an injective renaming ϕ which maps class, role, and individual names in $\mathcal{J}_1$ and η_1 to class, role, and individual names in $\mathcal{J}_2$ and η_2, respectively, such that $\phi(\mathcal{J}_1) = \mathcal{J}_2$ and $\phi(\eta_1) = \eta_2$.*

Example 3 (Isomorphic Justifications)

$$\begin{aligned}
\mathcal{J}_1 &= \{A \sqsubseteq B \sqcap \exists r.C, B \sqcap \exists r.C \sqsubseteq D\} \models A \sqsubseteq D \\
\mathcal{J}_2 &= \{E \sqsubseteq B \sqcap \exists s.F, B \sqcap \exists s.F \sqsubseteq D\} \models E \sqsubseteq D \\
\phi &= \{A \mapsto E, C \mapsto F, r \mapsto s\}
\end{aligned}$$

The relation $\approx_i$ is symmetric, reflexive and transitive, from which it follows that $\approx_i$ is an equivalence relation and thus partitions a set of justifications.

In the remainder of this paper, we may refer to the isomorphism defined above as *strict* isomorphism in order to distinguish it from the other equivalence relations.

3 Subexpression-Isomorphism

From the above definition of isomorphism it follows that only justifications which have the same number and types of axioms and subexpressions can be isomorphic. It is easy to see, however, that justifications can have a similar structure despite their use of different concept expressions, as demonstrated in Example 1. This motivates a notion of isomorphism which allows not only the mapping of concept names, but also that of complex subexpressions.

We introduce a *justification template* Θ, which functions as the *unifying justification* for the isomorphic justifications:

Definition 3 (Subexpression-Isomorphism) *Two justifications* $(\mathcal{J}_1, \eta_1)$, $(\mathcal{J}_2, \eta_2)$ *are* s-isomorphic $((\mathcal{J}_1, \eta_1) \approx_s (\mathcal{J}_2, \eta_2))$ *if there exists a justification* (Θ, η)*, called a* template*, and two injective substitutions* ϕ_1, ϕ_2*, such that*

1. $\Theta \models \eta$

2. $\phi_1(\Theta) = \mathcal{J}_1$ *and* $\phi_2(\Theta) = \mathcal{J}_2$

3. $\phi_1(\eta) = \eta_1$ *and* $\phi_2(\eta) = \eta_2$.

The mappings ϕ_1 *and* ϕ_2 *map class, role, and individual names in the template* (Θ, η) *to subexpressions of* $(\mathcal{J}_1,\ \eta_1)$ *and* $(\mathcal{J}_2, \eta_2)$*, respectively.*

Lemma 1 *1. The relation* $\approx_s$ *is reflexive, transitive and symmetric; it is therefore an equivalence relation and thus partitions a set of justifications.*

2. S-isomorphism is a more general case of strict isomorphism: $\mathcal{J}_1 \approx_i \mathcal{J}_2$ *implies* $\mathcal{J}_1 \approx_s \mathcal{J}_2$.

For a complete proof of Lemma 1 we refer the reader to the supporting materials page[4] for this paper.

4 Lemma-Isomorphism

While s-isomorphism covers a number of justifications that can be regarded as equivalent due to them requiring the same type of reasoning to reach the entailment, it only applies to justifications which have the same number of axioms. This does not take into account cases where the justifications differ only marginally in some subset, but where the general reasoning may be regarded as similar nonetheless. We therefore introduce the notion of *lemma-isomorphism*, which extends subexpression-isomorphism with the substitution of subsets of justifications through intermediate entailments, so-called *lemmas* [7]. The general motivation behind lemma-isomorphism is demonstrated by the following example:

Example 4

$$\mathcal{J}_1 = \{A \sqsubseteq \exists r.B, B \sqsubseteq C, \exists r.C \sqsubseteq D\} \models A \sqsubseteq D$$
$$\mathcal{J}_2 = \{A \sqsubseteq \exists r.B, B \sqsubseteq C, C \sqsubseteq D, D \sqsubseteq E, \exists r.E \sqsubseteq F\} \models A \sqsubseteq F$$

[4] http://owl.cs.manchester.ac.uk/research/publications/supporting-material/

It is straightforward to see that both $\mathcal{J}_1$ and $\mathcal{J}_2$ require the same type of reasoning from a human user. As the justifications only differ in the length of the atomic subsumption chains they contain, we can certainly consider them to be *similar* with respect to *some* similarity measure. However, the two justifications are not considered isomorphic with respect to the definitions for strict isomorphism or subexpression-isomorphism. We therefore introduce a new type of isomorphism which takes into account the fact that subsets of justifications can be replaced with intermediate entailments which follow from them.

4.1 Lemmas in OWL

Lemmas of OWL justifications have previously found use in the extension of justifications to *justification-oriented proofs* [7]. The following definitions introduce simplified variants of the definitions [7] of justification lemmas and lemmatisations. Please note that for the purpose of illustrating the effect of lemma-isomorphism, we will simplify the lemmatisations to a more specific type of lemmas in the next section.

Definition 4 (Lemma) *Let $\mathcal{J}$ be a justification for an entailment η. A* lemma *of $(\mathcal{J}, \eta)$ is an axiom λ for which there exists a subset $S \subseteq \mathcal{J}$ such that $S \models \lambda$. A* summarising *lemma of $(\mathcal{J}, \eta)$ is a lemma λ for which there exists an $S \subseteq \mathcal{J}$ such that $\mathcal{J} \setminus S \cup \{\lambda\} \models \eta$ for $S \models \lambda$.*

Definition 5 (Lemmatisation) *Let $(\mathcal{J}, \eta)$ be a justification, let $S_1 \dots S_k$ be subsets of $\mathcal{J}$, and let $\lambda_1 \dots \lambda_k$ be axioms satisfying $S_i \models \lambda_i$ for $i \in \{1, \dots, k\}$. Then the set $\mathcal{J}^{\Lambda} := (\mathcal{J} \setminus \bigcup S_i) \cup \bigcup \{\lambda_i\}$ for $i \in \{1, \dots, k\}$ is called a* lemmatisation *of $\mathcal{J}$ if $\mathcal{J}^{\Lambda} \models \eta$. A* summarising lemmatisation *comprises only summarising lemmas.*

4.2 Lemma-Isomorphism

Given the definitions for lemmatisations, we can now define lemma-isomorphism as an extension to subexpression-isomorphism:

Definition 6 (Lemma-isomorphism) *Two justifications $(\mathcal{J}_1, \eta_1)$, $(\mathcal{J}_2, \eta_2)$ are* ℓ-isomorphic *($(\mathcal{J}_1, \eta_1) \approx_{\ell} (\mathcal{J}_2, \eta_2)$) if there exist lemmatisations $\mathcal{J}_1^{\Lambda_1}, \mathcal{J}_2^{\Lambda_2}$ which are s-isomorphic: $\mathcal{J}_1^{\Lambda_1} \approx_s \mathcal{J}_2^{\Lambda_2}$.*

Lemma-isomorphism using arbitrary lemmas as defined above carries some undesirable properties: First, unlike the previously defined relations, it describes a relation which is *not transitive*. This isssue can be adressed by allowing only *summarising* lemmatisations. Second, the lemmatisation might differ strongly from the original justifications; in the most extreme case, the lemmatisation of a justification can be the entailment itself. We therefore have to introduce some *constraints* on the admissible lemmatisations in order to preserve the nature of the original justifications. In order to restrict the lemmatisations to justifications which do not differ too much from the original justification, we focus on substituting *obvious steps* with their lemmas.

4.3 Lemmatisations and Obvious Steps

The notion of *obvious proof steps* [10,5] describes how proof steps which are intuitively *obvious* can be replaced with their conclusion, thereby shortening the proof without omitting important information. We loosely base the lemma restriction on this obviousness and choose one such example of an obvious and frequently occurring constellation of axioms in OWL justifications, namely atomic subsumption chains.

In atomic subsumption chains of the type $A_0 \sqsubseteq A_1, A_1 \sqsubseteq A_2 \ldots A_{n-1} \sqsubseteq A_n$ only the relation between the subconcept A_0 in the first axiom and the superconcept A_n in the last axiom are relevant for understanding the subsumption chain; i.e. the step from the subconcept to the final superconcept is *obvious*. We can say that it is only important to understand *that* there is a connection between the subconcept and the final superconcept, but we do not need to know *what* this connection is. Therefore, it seems reasonable to substitute the chain with its conclusion in the form of a single axiom $A_0 \sqsubseteq A_n$. Please note that it is possible for such a substitution to generate a non-summarising lemma; therefore, we will only allow *summarising lemmatisations based on atomic subsumption chains.*

Atomic subsumption chains represent only one of many examples of such lemmatisations which preserve both transitivity and the original style of the justification. For the purpose of introducing lemma-isomorphism as an equivalence relation in this paper, we focus on this particular type of lemmatisations, as it captures a frequently occurring pattern in OWL justifications.

5 Diversity of Reason in the NCBO BioPortal Ontologies

5.1 Test Corpus

We performed a survey of equivalence relations in OWL- and OBO-ontologies from the NCBO BioPortal.[5] The purpose of this study was to determine the prevalence of the different types of isomorphism across an independently motivated (as opposed to hand selected) corpus of OWL ontologies used in practice.

At the time of downloading (January 2012), the BioPortal listed 278 OWL- and OBO-ontologies, of which 241 could be downloaded, merged with their imports, and serialised as OWL/XML. 15 of those ontologies could not be processed in the given time frame of 30 minutes using the selected reasoner, and another 25 did not contain any relevant entailments (direct subsumptions between named classes). For the remaining 201 ontologies, we computed justifications for all entailments with a maximum of 500 justifications per entailment. Self-supporting entailments and self-justifications were excluded from the survey, which led to the discarding of further ontologies.

The final corpus of justifications consisted of 6,744 justifications from 83 ontologies, covering a very broad spectrum of sizes and complexity. Half of the ontologies had less than 1000 named concepts and axioms, with the other half

[5] http://bioportal.bioontology.org/

reaching a maximum of 13,959 concepts and 70,015 axioms. Likewise, the expressivity of the ontologies ranged from $\mathcal{AL}$ to several highly expressive samples in $\mathcal{SROIQ}$. A detailed listing of all surveyed ontologies alongside the study results is available online.[6]

5.2 Isomorphism on the Entailment Level

We first analysed how the equivalence relations affected the set of justifications for a *single* entailment. For this purpose we focused exclusively on those 39 ontologies in the corpus which produced entailments with multiple justifications. 5,647 justifications were computed for the 3,264 entailments of those ontologies (including those entailments which had only 1 justification).

Strict Isomorphism On average, an entailment in the reduced corpus has 1.7 justifications, with a maximum of 122 justifications for an entailment from the *Orphanet Ontology of Rare Diseases*. Strict isomorphism shows a significant reduction by 23.6% to an average of 1.3 templates per entailment. Overall, however, only few ontologies are visibly affected by this reduction: In 11 ontologies, an average of 3 justifications for an individual entailment is covered by a single template, in 13 ontologies a template covers an average of 2 justifications, and in the remaining 15 ontologies strict isomorphism does not affect the numbers of justifications per entailment.

Of those 11 ontologies which do show some significant reduction, entailments of the Orphanet and *Cognitive Atlas* ontology reveal the most striking regularities: The 122 justifications from the Orphanet ontology were covered by only 2 templates, with 61 justifications each:

$$\Theta_1 = \{A \sqsubseteq \exists r.B, Domain(r,\ C)\} \models A \sqsubseteq C$$
$$\Theta_2 = \{A \sqsubseteq \exists s.B, s \sqsubseteq r, Domain(r,\ C)\} \models A \sqsubseteq C$$

This pattern is repeated by a large number of entailments across the Orphanet ontology; as we will see in the next section, almost all entailments in this ontology have justifications which are covered by these two templates.

S-Isomorphism Subexpression-isomorphism affects the justifications of only 12 of the 3,264 entailments. Most of these stem from the *Bleeding History Phenotype* ontology, where the template Θ_1 also covers justifications of the type $\{A \sqsubseteq \exists r.(B \sqcup D), Domain(r,\ C)\}$, i.e. they contain a disjunction instead of the atomic class name B as the filler of the existential restriction.

L-Isomorphism Similarly, lemma-isomorphism only affects 39 entailments, with the most notable effects in the *Human Developmental Anatomy* ontology, where justifications comprising of atomic subsumption chains of lengths 2 and 3, respectively, are covered by a single template.

[6] See footnote 4.

5.3 Isomorphism Across Multiple Entailments

Strict Isomorphism When applied to the justifications for *all* entailments of the individual ontologies, strict isomorphism drastically reduces the number of justifications from an average of 81.3 ($\sigma = 185.5$) justifications per ontology to 10.5 ($\sigma = 18.0$) templates for equivalent justifications. The mean number of justifications per template is 7.7 ($\sigma = 41.7$), which means that in each ontology nearly 8 justifications have an identical structure. This effect is highly visible in the Orphanet ontology, where the above template Θ_1 covers 901 (of 1139) justifications for *distinct* entailments.

S-Isomorphism The reduction from strict isomorphism to s-isomorphism is less drastic than the difference between the main pool and the non-isomorphic pool. The justifications of the 83 ontologies are reduced from an average of 81.3 justifications to 8.8 templates ($\sigma = 13.1$), which is a reduction by 1.7 templates compared to strict isomorphism. An average of 9.2 justifications ($\sigma = 46.6$) in an ontology share the same template. Surprisingly, the majority of ontologies (67) does not show any difference between strict isomorphism and s-isomorphism. Only 2 ontologies, the *Lipid Ontology* and Bleeding History Phenotype, are significantly affected by s-isomorphism, with a reduction from 118 to 13 templates (an 89% reduction from strict isomorphism) and 32 to 14 templates (46.2% reduction from strict isomorphism), respectively.

L-Isomorphism As with s-isomorphism, the effects of ℓ-isomorphism are not as significant as the first reduction through strict isomorphism. The justifications are further reduced to an average of 7.4 templates per ontology ($\sigma = 11.4$), with 11 justifications per template ($\sigma = 51.5$). Still, 35 of the 83 ontologies show at least a minor difference between s-isomorphism and ℓ-isomorphism, which indicates that they contain at least 1 atomic subsumption chain. L-isomorphism reduces the 106 justifications generated for the *Cereal Plant Gross Anatomy* ontology to only 14 templates, compared to 29 templates for s-isomorphism.

5.4 Similarities Across Multiple Ontologies

Strict Isomorphism When applied across all justifications from the corpus, strict isomorphism reduces the corpus from 6,744 justifications to only 614 templates, a reduction to only 9.1% of the original set of justifications. On average, 11 justifications share the same template, with the most frequent template occurring 1,603 times across 18 different ontologies (that is, in about a fifth of all ontologies); this template is of the same form as the Orphanet Ontology described above.

S-Isomorphism Subexpression-isomorphism reduces the corpus from 6,744 to 456 templates (6.8% of the corpus), which is a further reduction by 25.7% compared to the 614 templates for strict isomorphism. The most frequent templates in terms of number of justifications and prevalence across all ontologies are the same as for strict isomorphism, with numbers differing only slightly.

L-Isomorphism Finally, lemma-isomorphism reduces the 6,744 justifications to a mere 384 templates, which is an overall reduction of 94.3%, and a further reduction by 15.8% compared to subexpression-isomorphism. The effect of lemma-isomorphism is visible when we look at the most prevalent justification, an atomic subsumption chain of size 2, which occurs in 44 (compared to previously 37) ontologies. This chain represents all 701 atomic subsumption chains of differing sizes that can be found in the corpus.

5.5 Summary

The results of our survey indicate that the effects of the three equivalence relations vary strongly between the ontologies in the corpus. In contrast to strict isomorphism, subexpression- and lemma-isomorphism have almost no effect on the justifications for *individual* entailments. For multiple entailments, however, some ontologies show a clear reaction to s- or ℓ-isomorphism. Across the corpus, the logical diversity could be shown to be significantly smaller than the number of justifications would suggest, as lemma-isomorphism reduced the over 6000 justifications to only around 600 distinct templates.

6 An Application Scenario

The methods proposed in this paper were motivated by an example from the well-known *Pizza* tutorial ontology.[7] An example entailment for this ontology is $\mathsf{Fiorentina} \sqsubseteq \mathsf{InterestingPizza}$, which has over 200 justifications. An ontology engineer wanting to understand why this entailment holds, for example because it is considered incorrect, would have to go through a list of several hundred justifications, inspecting each one and deciding which axiom to modify or remove in order to 'break' the entailment.

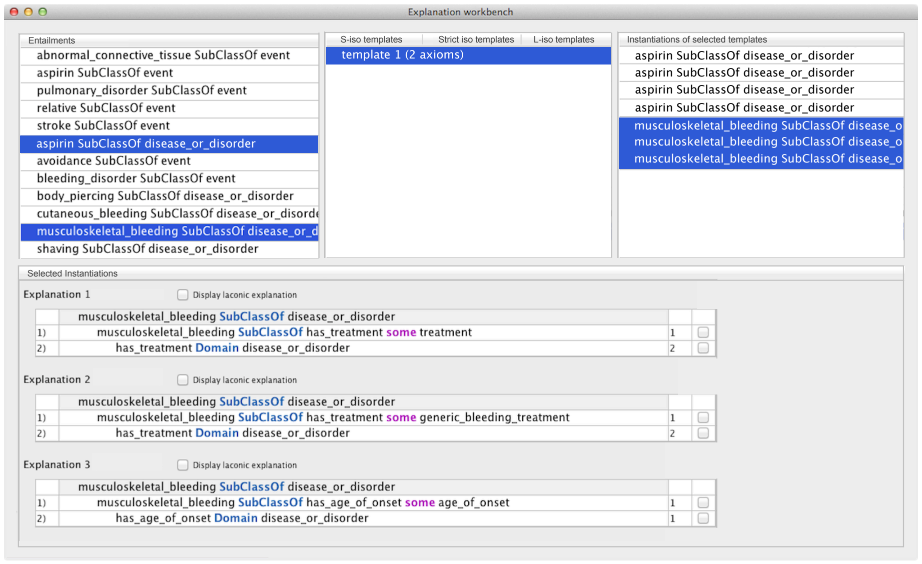
Closer inspection, however, reveals significant similarities between the justifications for this entailment: All justifications are of the form

$$\{\mathsf{S}_1, \mathsf{S}_2, \mathsf{InterestingPizza} \equiv \mathsf{Pizza} \sqcap (\geqslant 3\ \mathsf{hasTopping}.\top)\}$$

where S_1 is one of several axiom sets entailing that $\mathsf{Fiorentina} \sqsubseteq \mathsf{Pizza}$, and S_2 a set of axioms entailing that $\mathsf{Fiorentina} \sqsubseteq \geqslant 3\ \mathsf{hasTopping}.\top$, which originates from the fact that the $\mathsf{Fiorentina}$ pizza is defined to have six disjoint toppings. While the large number of justifications may seem daunting at first, once the structural similarities have been spotted, understanding the different reasons why the entailment holds requires significantly less effort—both mentally, and in terms of a 'click-count'.

Integrating the proposed equivalence relations into a user interface could support users in spotting these patterns. In the case of the *Pizza* ontology, we can apply techniques that make use of both strict isomorphism and lemma-isomorphism: Strict isomorphism may be used to group those justifications which

[7] http://owl.cs.manchester.ac.uk/tutorials/protegeowltutorial/


Explanation workbench

Entailments
abnormal_connective_tissue SubClassOf event
aspirin SubClassOf event
pulmonary_disorder SubClassOf event
relative SubClassOf event
stroke SubClassOf event
aspirin SubClassOf disease_or_disorder
avoidance SubClassOf event
bleeding_disorder SubClassOf event
body_piercing SubClassOf disease_or_disorder
cutaneous_bleeding SubClassOf disease_or_disorde
musculoskeletal_bleeding SubClassOf disease_or_d
shaving SubClassOf disease_or_disorder

S-iso templates | Strict iso templates | L-iso templates
template 1 (2 axioms)

Instantiations of selected templates
aspirin SubClassOf disease_or_disorder
aspirin SubClassOf disease_or_disorder
aspirin SubClassOf disease_or_disorder
aspirin SubClassOf disease_or_disorder
musculoskeletal_bleeding SubClassOf disease_o
musculoskeletal_bleeding SubClassOf disease_o
musculoskeletal_bleeding SubClassOf disease_o

Selected Instantiations

Explanation 1 ☐ Display laconic explanation

	musculoskeletal_bleeding SubClassOf disease_or_disorder		
1)	musculoskeletal_bleeding SubClassOf has_treatment some treatment	1	☐
2)	has_treatment Domain disease_or_disorder	2	☐

Explanation 2 ☐ Display laconic explanation

	musculoskeletal_bleeding SubClassOf disease_or_disorder		
1)	musculoskeletal_bleeding SubClassOf has_treatment some generic_bleeding_treatment	1	☐
2)	has_treatment Domain disease_or_disorder	2	☐

Explanation 3 ☐ Display laconic explanation

	musculoskeletal_bleeding SubClassOf disease_or_disorder		
1)	musculoskeletal_bleeding SubClassOf has_age_of_onset some age_of_onset	1	☐
2)	has_age_of_onset Domain disease_or_disorder	1	☐

Fig. 1: Screenshot of a justification template exploration tool

have identical subexpressions modulo the names of the different toppings. L-isomorphism (extended by additional types of 'obvious' proof steps) could further group those axiom sets which lead to the same lemmas. A basic interaction mechanism for navigation such lemmatisations has been suggested in the work on *justification-based proofs* [7]. Going beyond the example of the *Pizza* ontology, s-isomorphic justifications could easily be highlighted by covering up irrelevant expressions similar to the strike-out techniques for superfluous expressions implemented in the Swoop ontology editor [12]. This would prevent users from getting distracted by complex expressions, thus allowing them to focus on understanding the relevant axioms and expressions in a set of justifications.

The second task for which our notions of isomorphism may be useful is the exploration and understanding of an ontology, without focusing on a specific entailment. In this case, the user could be offered a browser-type interface as the one shown in Figure 1 (displaying entailments from the Bleeding History Phenotype ontology). The browser-style exploration tool consists of three top panels, which show a list of entailments, a list of the formal templates which cover the justifications for the selected entailment(s), and a list of material instantiations of the selected template (identified by the entailment they stand for). The bottom panel displays the selected instantiations or templates. A user seeking to understand the structure of an ontology could gain a high-level view of the ontology by selecting a set of entailments which then displays the set of their justification templates and respective instantiations of those templates. Entailments which share a template (or a number of templates) then highlight regularities in the axiom structure of the ontology, as well as the prevalence of such regularities.

7 Conclusions

In this paper, we introduced new types of equivalence relations between OWL justifications, subexpression-isomorpism and lemma-isomorphism. We demonstrated how a seemingly diverse corpus of justifications from the NCBO BioPortal could be reduced by over 90% to a much smaller set of non-isomorphic justifications. We have found that, surprisingly, most justifications are in fact strictly isomorphic, with only a few ontologies being affected by the other equivalence relations.

Future work will involve exploring further notions of obvious proof steps in order to extend lemma-isomorphism beyond atomic subsumption chains. We will also consider the issue of overlapping chains, i.e. subsumption chains which lead to non-summarising lemmas. Finally, we aim to fully implement the proposed tool which orders and groups justifications based on their isomorphism relations, and conduct user studies that investigate the usefulness of the tool for various tasks in the ontology development process.

References

1. Baader, F., Küsters, R., Borgida, A., McGuinness, D.L.: Matching in description logics. J. of Logic and Computation 9(3), 411–447 (1999)
2. Baader, F., Morawska, B.: Unification in the description logic EL. In: Proc. of RTA-09. pp. 350–364 (2009)
3. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic $\mathcal{EL}^+$. In: Proc. of KI-07. pp. 52–67 (2007)
4. Bail, S., Parsia, B., Sattler, U.: The justificatory structure of OWL ontologies. In: Proc. of OWLED-10 (2010)
5. Davis, M.: Obvious logical inferences. In: Proc. of IJCAI-81. pp. 530–531 (1981)
6. Horridge, M., Bail, S., Parsia, B., Sattler, U.: The cognitive complexity of OWL justifications. In: Proc. of ISWC-11 (2011)
7. Horridge, M., Parsia, B.: From justifications to proofs for entailments in OWL. In: Proc. of OWLED-09 (2009)
8. Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in OWL. In: Proc. of ISWC-08. pp. 323–338 (2008)
9. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. of KR-06 (2006)
10. Johnson-Laird, P.N.: Mental models in cognitive science. Cognitive Science 4(1), 71–115 (1980)
11. Kalyanpur, A., Parsia, B., Cuenca Grau, B.: Beyond asserted axioms: Fine-grain justifications for OWL-DL entailments. In: Proc. of DL-06 (2006)
12. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca Grau, B., Hendler, J.: Swoop: A web ontology editing browser. J. of Web Semantics 4(2), 144–153 (2006)
13. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in OWL ontologies. J. of Web Semantics 3(4), 268–293 (2005)
14. Meyer, T., Moodley, K., Varzinczak, I.: First steps in the computation of root justifications. In: Proc. of ARCOE-10 (2010)
15. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proc. of IJCAI-03. pp. 355–362 (2003)

Debugging Taxonomies and their Alignments: the ToxOntology - MeSH Use Case

Valentina Ivanova[1], Jonas Laurila Bergman[2], Ulf Hammerling[3], Patrick Lambrix[1]

(1) Department of Computer and Information Science,
and Swedish e-Science Researche Centre, Linköping University, SE-581 83 Linköping, Sweden
(2) Division of Information Technology, National Food Agency, SE-75126 Uppsala, Sweden
(3) Department of Risk Benefit Assessment, National Food Agency, SE-75126 Uppsala, Sweden

Abstract. As part of an initiative to facilitate adequate identification and display of substance-associated health effects a toxicological ontology - ToxOntology - was created. Further, an alignent with MeSH was accomplished to obtain an indirect index to the scientific literature.
To arrive at satisfactory results in the semantically-enabled applications, high-quality ontologies and alignments are both necessary. A key step towards high quality in this area is debugging the ontologies and their alignments. In this paper we present an experience report on the debugging of ToxOntology and MeSH as well as an alignment.

1 Introduction

Toxicology information, publicly available via Internet, has grown immensely over the last decade and represents a major fundament to risk assessment in a range of regulatory applications, including that of food toxicology. This corpus is commonly referred to as the Internet-based toxicology landscape [21, 10, 17]. The accordingly deposited information is, however, heterogeneous i.e. appears in various forms and formats and is distributed across a rich variety of databases. Several harmonization initiatives have, however, been launched to help extracting such information from disparate sources, typified by the construction of Internet portals (e.g. Toxnet and eChemPortal) and data format standardization [20, 26]. Moreover, the demarcation between data holding classical toxicology actions of substances and that of their general biological activity has become less sharp in recent years. Notably, the ToxCast and Tox21 initiatives have provided gargantuan amounts of data - freely available through the PubChem repository - encompassing results from a wide range of *in vitro* biological assay outputs on noxious chemicals, and the Computational Toxicogenomics Database merges molecular data on chemical health effects at various levels of resolution [18, 1, 22]. Actually, even interaction-type data has recently witnessed exploitation in computational toxicology [8, 2]. Moreover, the OpenTox project, funded by the 7th EU Framework Programme for research, aims at facilitating informatics work in toxicology, through providing an interoperable and standardized framework to support predictive toxicology [4]. Nonetheless, exhaustive toxicology data search and crosswise comparison can still be a cumbersome undertaking.

As part of a slightly broader initiative to facilitate the identification of adequate substance-associated health effects a toxicological ontology - ToxOntology - was created within an informatics system development at the Swedish National Food Agency (NFA). It is inspired by and incorporates several toxicology endpoints of the REACH chemical legislature framework, on which a considerably larger endpoints ontology has been built, as developed within the OpenTox community [16, 25]. While OpenTox vocabularies are mainly designed for advancing predictive toxicology - especially QSAR modeling - the purpose of ToxOntology, however, is to support the identification and presentation of health effects associated with (chemical) substances, as appearing in databases and the scientific literature. Terms and architecture of ToxOntology were created manually by expert toxicologists using various relevant regulatory documents as well as scientific papers in the field. ToxOntology is used in an in-house tagging service to mark textual records where existing classification systems lack coverage, and in an ontology-based text mining application. It is supported by a navigation tool for accessing databases and literature.

Further, the scientific literature is a major source of toxicology information not yet being curated and rendered available in databases. A key source of such documentation is MEDLINE, using Medical Subject Headings (MeSH, [5]) as a classification system. Although the previously mentioned tagging service could be used here for indexing articles relating to a substance of interest, a more precise connection to an already curated index was desired, implicating a need of an alignment [3] between ToxOntology and MeSH.

To obtain high-quality results in semantically-enabled applications (such as the ontology-based text mining and search applications), high-quality ontologies and alignments are both necessary. A key step towards higher quality is to debug the ontologies and their alignments. In this paper we present an experience report on the debugging of ToxOntology and MeSH as well as an alignment. In section 2 we briefly describe ToxOntology and MeSH, as well as the ontology alignment and the ontology debugging systems that were used. Section 3 describes the actual debugging experience, including the creation of an initial alignment of ToxOntology and MeSH, the detection of possible defects using RepOSE [11], two independent repairing sessions - manual and using RepOSE, as well as an experiment using a non-validated initial alignment. The paper concludes in section 4.

2 Background

ToxOntology. ToxOntology is an OWL2 ontology, encompassing 263 concepts and 266 asserted is-a relations. The ontology has ten main axes (top concepts) including Toxic effect, Route of exposure and Time of exposure. All concepts have human readable labels and synonyms attached. ToxOntology appeared after a merge of classification systems covering concepts within toxicology used by ACToR [9] and an implementation of the OpenTox API [6]. The merge was further refined and expanded manually by toxicology experts at the NFA, end-users of ToxOntology. The overall design principle can be summarized as follows: broad enough to cover almost any aspect of interest in the field and at the same time small enough to become an interactive tool in users' daily search of toxicology information.

MeSH. MeSH is a thesaurus of the National Library of Medicine (NLM). It consists of sets of terms naming descriptors in a 12-level hierarchical structure. The 2011 version of MeSH contains 26,142 descriptors. MeSH is used by NLM largely for indexing PubMed [19]. As MeSH contains many descriptors not related to the domain of toxicology, we used parts from the Diseases [C], Analytical, Diagnostic and Therapeutic Techniques and Equipment [E] and Phenomena and Processes [G] branches of MeSH. The resulting ontology contained 9,878 concepts and 15,786 asserted is-a relations. A Java program was written to parse (using the SAX parser) the XML file, filter the selected elements and create the OWL file (using Jena2.1). We note that the MeSH hierarchy is not based on subsumption relations only, and thus interpreting all structural relations as is-a relations, may lead to unintended results.

Ontology alignment system - SAMBO/KitAMO. Our ontology alignment system SAMBO (e.g. [14, 24, 12]) is based on the framework defined in [12] and implements different strategies for preprocessing, matching, combining and filtering. We briefly discuss the strategies that were used in this use case. We did not use preprocessing strategies to reduce the search space. Matchers calculate similarity values between terms. As matchers we used *TermBasic* (linguistic approach), *TermWN* (approach using WordNet [27]), *UMLSM* (approach using domain knowledge - UMLS [23]), and *NaiveBayes* (instance-based approach using scientific literature). The results of the matchers can be combined in different ways. In this use case we used the maximum-based combination strategy, which returns as final similarity value between terms, the maximum value of the similarity values computed by the individual matchers. Further, we used the single threshold filtering strategy, that retains pairs of terms with a similarity value equal to or higher than a given threshold value as mappings suggestions. The mapping suggestions should then be validated by a domain expert. KitAMO [15] is a tool for evaluating and analyzing ontology alignment strategies and their combinations. The tool covers the non-interactive part of the general framework for aligning ontologies. We have used the KitAMO tool with the SAMBO strategies mentioned above, thereby allowing us to store and analyze results from different runs of the algorithms.

Ontology debugging system - RepOSE. RepOSE (version as described in [11]) is a logic-based tool for debugging is-a structure within and mappings between taxonomies. It covers the detection and repairing of defects. It handles defects regarding missing as well as wrong is-a structure, and defects regarding missing and wrong equivalence and is-a mappings. It is based on the framework for debugging ontologies shown in Figure 1. The debugging workflow consists of 6 phases, where the first two phases are for the detection and validation of possible defects, and the last four are for the repairing. The input is a network of ontologies. The output is the set of repaired ontologies and alignments.

In the current version of RepOSE, the detection of defects uses information inherent in the network consisting of the taxonomies and the alignments. In **Phase 1** the system

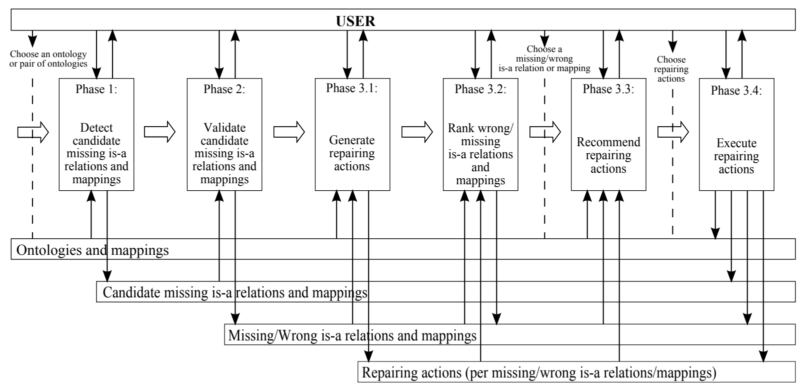

USER
Choose an ontology or pair of ontologies
Phase 1: Detect candidate missing is-a relations and mappings
Phase 2: Validate candidate missing is-a relations and mappings
Phase 3.1: Generate repairing actions
Phase 3.2: Rank wrong/ missing is-a relations and mappings
Choose a missing/wrong is-a relation or mapping
Phase 3.3: Recommend repairing actions
Choose repairing actions
Phase 3.4: Execute repairing actions
Ontologies and mappings
Candidate missing is-a relations and mappings
Missing/Wrong is-a relations and mappings
Repairing actions (per missing/wrong is-a relations/mappings)

Fig. 1. Debugging workflow [11].

computes for every taxonomy the is-a relations that can be derived from the network but not from the taxonomy alone. These are called *candidate missing is-a relations* (CMIs). Similarly, it computes for every pair of taxonomies and their alignment the mappings that can be logically derived from the network but not from the taxonomies and their alignment alone. These are called *candidate missing mappings* (CMMs). As these CMIs and CMMs may be derived using erroneous information in the network, a domain expert is needed to validate and classify them into missing is-a relation, wrong is-a relation, missing mapping or wrong mapping (**Phase 2**). The CMIs and CMMs are shown to the domain expert using arrows together with their justification[1]. Related items are shown together. The user can validate by clicking the arrows and toggle the label to ’W’ or ’M’ (e.g. Figure 2). There is also a recommendation algorithm that uses external knowledge. We note that each of the validated CMIs and CMMs gives rise to a debugging opportunity. Missing is-a relations and mappings should be repaired by adding information to taxonomies or alignments. Wrong is-a relations and mappings are repaired by removing information from taxonomies or alignments.

Ontologies and alignments are repaired one by one. For the selected taxonomy or for the selected alignment and its pair of taxonomies, a user can choose to repair the missing or the wrong is-a relations/mappings **(Phase 3.1-3.4)**. Although the algorithms for repairing are different for missing and wrong is-a relations/mappings, the repairing goes through the phases of generation of repairing actions, the ranking of is-a relations/mappings, the recommendation of repairing actions and finally, the execution of repairing actions. In **Phase 3.1** repairing actions are generated. For wrong is-a relations and mappings, the repairing actions are is-a relations or mappings to remove. For each wrong is-a relation and mapping the justifications in the network are computed. The defect can be repaired by removing at least one is-a relation or mapping in each jus-

[1] A justification for an is-a relation or mapping can be seen as an explanation for why this is-a relation or mapping is derivable from the network. It is a minimal set of is-a relations and mappings that allows for the derivation of the given is-a relation or mapping. For a formal definition, see e.g. [11, 7].

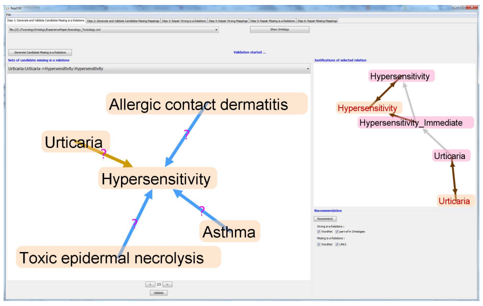

RepOSE
File
Step 1: Generate and Validate Candidate Missing is-a Relations
Step 2: Generate and Validate Candidate Missing Mappings
Step 3: Repair Wrong is-a Relations
Step 4: Repair Wrong Mappings
Step 5: Repair Missing is-a Relations
Step 6: Repair Missing Mappings
file:///C:/ToxicologyOntology/ExperiencePaper/toxicology_Toxicology.owl
Show Ontology
Generate Candidate Missing is-a Relations
Validation started ....
Sets of candidate missing is-a relations
Urticaria:Urticaria->Hypersensitivity:Hypersensitivity
Allergic contact dermatitis
Urticaria
Hypersensitivity
Asthma
Toxic epidermal necrolysis
Justifications of selected relation
Hypersensitivity
Hypersensitivity
Hypersensitivity_Immediate
Urticaria
Urticaria
Recommendation
Recommend
Wrong is-a Relations :
WordNet
part-of in Ontologies
Missing is-a Relations :
WordNet
UMLS

Fig. 2. Generating and validating CMIs.

tification. RepOSE shows for each wrong is-a relation or mapping the justifications as directed graphs (Figure 3). The domain expert can repair by choosing edges in the graph and commit to removing them. For each missing is-a relation or mapping, a Source set and a Target set are computed.[2] It is guaranteed that when an is-a relation/mapping is added between any element in the Source set and any element in the Target set, the defect is repaired. The algorithm also guarantees solutions adhering to a number of heuristics [13]. The Source and Target sets are displayed in two panels to the domain expert (together with the justification of the missing is-a relation or mapping) allowing the user to conveniently repair defects by selecting elements in the panels (Figure 4). In general, there will be many is-a relations/mappings needing repairment and some of them may be easier to embark on such as those with few repairing actions. We therefore rank them with respect to the number of possible repairing actions **(Phase 3.2)**. After this, the user can select an is-a relation/mapping to repair and choose among possible repairing actions. To facilitate this process, we developed methods to guide the user by means of advised repairing actions **(Phase 3.3)**. Once the user decides on repairing actions, the chosen repairing actions are then applied to the relevant taxonomies and alignments and the consequences are computed **(Phase 3.4)**. We also note that the user can switch between different ontologies and phases at any time during the process.

3 Debugging ToxOntology, MeSH and their alignment

3.1 Aligning ToxOntology and MeSH

As an alignment of ToxOntology and MeSH was deemed necessary, and as RepOSE uses an alignment in the detection phase of defects, the first step of our process was to create an initial alignment between ToxOntology and MeSH. Moreover, due to a preference for an as complete as possible, high-quality alignment, preprocessing to reduce the search space was excluded from the procedure; we used different types of matchers; and as combination strategy we used the maximum-based strategy. We generated the similarity values for all pairs of terms. Further, we used single threshold filtering

[2] Essentially, for missing is-a relation a $\rightarrow$ b, Source(a,b) = super-concepts(a) $\setminus$ super-concepts(b) and Target(a,b) = sub-concepts(b) $\setminus$ sub-concepts(a).

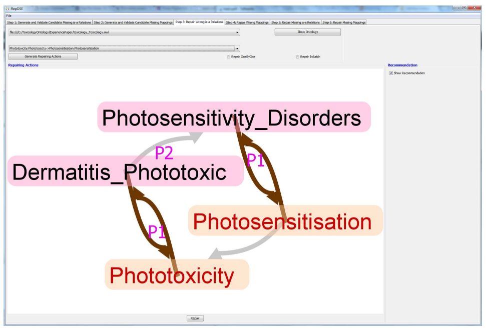


Fig. 3. Repairing wrong is-a relations.

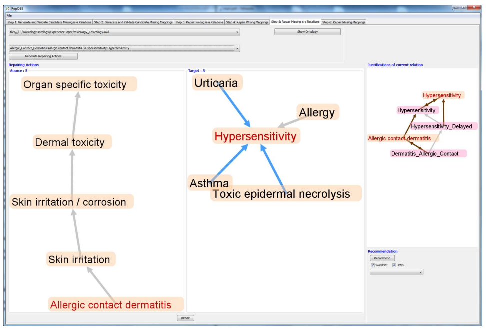


Fig. 4. Repairing missing is-a relations.

similarity value	suggestions	equivalence	ToxOntology is-a MeSH	MeSH is-a ToxOntology	related	wrong
≥ 0.8	41	29	2	2	1	7
$\geq 0.5, < 0.8$	419	9	18	31	42	319
$\geq 0.4, < 0.5$	906	2	21	14	83	786
$\geq 0.35, < 0.4$	146	1	2	2	117	24

Fig. 5. Validation of mapping suggestions - initial alignment.

with threshold 0.35 for the filtering strategy. These choices would lead to a high recall, although there would be many mapping suggestions to validate.

During the validation phase the domain expert classified the mapping suggestions into: equivalence mapping, is-a mapping (ToxOntology term is-a MeSH term and MeSH term is-a ToxOntology term), related terms mapping and wrong mapping. The mapping suggestions were shown to the domain expert in different steps based on the similarity values. The results are summarized in Figure 5. The validated alignment consists of 41 equivalence mappings, 43 is-a mappings between a ToxOntology term and a MeSH term, 49 is-a mappings between a MeSH term and a ToxOntology term and 243 related terms mappings. Further, there is information about 1,136 wrong mappings.

3.2 Debugging using validated alignment

It was not considered feasible to identify defects manually. Therefore, we used the detection mechanisms of RepOSE. RepOSE computed CMIs, which were then validated by domain experts. As there initially were only 29 CMIs, we decided to repair the ontologies and their alignment independently in two ways. First, the CMIs and their justifications were given to the domain experts who manually repaired the ontologies and their alignment. Second, the repairing mechanisms of RepOSE were used. A summary of the changes in the alignment and in ToxOntology due to the debugging sessions are summarized in Figure 6 columns 'original alignment' and 'final alignment'[3], and Fig-

[3] The final alignment contains changes from the two debugging sessions and is the one that is now used.

ToxOntology	MeSH	original alignment	final alignment	final alignment manual	final alignment RepOSE
metabolism	metabolism	$\equiv$	$\rightarrow$	$\rightarrow$	removed $\leftarrow$
photosensitisation	photosensitivity disorders	$\equiv$	R	R	removed $\leftarrow$, $\rightarrow$
phototoxicity	dermatitis phototoxic	$\equiv$	R	R	removed $\leftarrow$, $\rightarrow$
inhalation	administration inhalation	$\equiv$	W	W	removed $\leftarrow$, $\rightarrow$
urticaria	urticaria pigmentosa	$\leftarrow$	W	W	removed $\leftarrow$
autoimmunity	diabetes mellitus type 1	$\leftarrow$	R	R	removed $\leftarrow$
autoimmunity	hepatitis autoimmune	$\leftarrow$	R	R	removed $\leftarrow$
autoimmunity	thyroiditis autoimmune	$\leftarrow$	R	R	removed $\leftarrow$
gastrointestinal metabolism	carbohydrate metabolism	$\leftarrow$	W	W	removed $\leftarrow$
gastrointestinal metabolism	lipid metabolism	$\leftarrow$	W	W	removed $\leftarrow$
cirrhosis	fibrosis	$\equiv$	R	R	removed $\leftarrow$, $\rightarrow$
cirrhosis	liver cirrhosis	$\leftarrow$	$\equiv$	$\equiv$	-
metabolism	biotransformation	$\leftarrow$	$\equiv$	$\equiv$	-
metabolism	carbohydrate metabolism	$\leftarrow$	W	W	-
metabolism	lipid metabolism	$\leftarrow$	W	W	-
hepatic porphyria	porphyrias	$\equiv$	$\rightarrow$	W	removed $\leftarrow$
hepatic porphyria	drug induced liver injury	$\rightarrow$	R	-	removed $\rightarrow$

Fig. 6. Changes in the alignment (equivalence mapping ($\equiv$), ToxOntology term is-a MeSH term ($\rightarrow$), MeSH term is-a ToxOntology term ($\leftarrow$), related terms (R), wrong mapping (W)).

ure 7 column 'final', respectively. There are also 5 missing is-a relations for MeSH. In the remainder of this subsection we describe the detection and repairing in more details and compare the manual repairing with the repairing using RepOSE.

Detection using RepOSE - first run. As input to RepOSE we used ToxOntology and MeSH as discussed in section 2. Further, we used the validated part of the alignment discussed in section 3.1, that contains the 41 equivalence mappings, the 43 is-a mappings between a ToxOntology term and a MeSH term and the 48 is-a mappings between a MeSH term and a ToxOntology term.[4]

RepOSE generated 12 non-redundant CMIs for ToxOntology (34 in total) of which 9 were validated by the domain experts as missing and 3 as wrong. For MeSH, RepOSE generated 17 non-redundant CMIs (among which 2 relations represented one equivalence relation - 32 CMIs in total) of which 5 were validated as missing and the rest as wrong.

Manual repair. The domain experts focused on repairment of ToxOntology and the alignment. Regarding the 9 missing is-a relations in ToxOntology, these were all added to the ontology. Further, another is-a relation, *asthma* $\rightarrow$ *respiratory toxicity*, was added,

[4] The related term mappings cannot be used in logical derivation related to the is-a structure of the ontologies and are therefore not included in the alignment used in RepOSE.

Added is-a relations	final	manual	RepOSE
absorption → physicochemical parameter	Yes	Yes	Yes
hydrolysis → metabolism	Yes	Yes	Yes
toxic epidermal necrolysis → hypersensitivity	Yes	Yes	Yes
urticaria → hypersensitivity	Yes	Yes	Yes
asthma → hypersensitivity	Yes	Yes	Yes
asthma → respiratory toxicity	Yes	Yes	No
allergic contact dermatitis → hypersensitivity	Yes	Yes	Yes
subcutaneous absorption → dermal absorption	Yes	Yes	Yes
oxidation → metabolism	Yes	Yes	Yes
oxidation → physicochemical parameter	Yes	Yes	Yes

Fig. 7. Changes in the structure of ToxOntology.

in addition to *asthma* → *hypersensitivity*, based on an analogy of this case with the already existing *urticaria* → *dermal toxicity* and added *urticaria* → *hypersensitivity*. This is summarized in Figure 7 column 'manual'. The domain experts also removed two asserted is-a relations (*asthma* → *immunotoxicity* and *subcutaneous absorption* → *absorption*) for reasons of redundancy. These is-a relations are valid and they are derivable in ToxOntology.

The wrong is-a relations for MeSH and ToxOntology were all repaired by removing mappings in the alignment (Figure 6 column 'final alignment manual'). In 5 cases a mapping was changed from equivalence or is-a into related. In one of the cases (concerning *cirrhosis* in ToxicOntology and *fibrosis* and *liver cirrhosis* in MeSH) a further study also led to the change of *cirrhosis* ← *liver cirrhosis* into *cirrhosis* ≡ *liver cirrhosis*.

The wrong is-a relations involving *metabolism* in ToxOntology, invoked a deeper study of the use of this term in ToxOntology and in MeSH. The domain experts concluded that the ToxOntology term *metabolism* is equivalent to the MeSH term *biotransformation* and a subconcept of the MeSH term *metabolism*. This observation led to a repair of the mappings related to *metabolism*.

Further, some mappings were changed from an equivalence or is-a mapping to a wrong mapping.[5] In these cases (e.g. between *urticaria* in ToxOntology and *urticaria pigmentosa* in MeSH) the terms were syntactically similar and were initially validated wrongly during the alignment phase.

Repairing using RepOSE. For the 3 wrong is-a relations for ToxOntology and the 12 wrong is-a relations for MeSH, the justifications were shown to the domain experts. The justifications for a wrong is-a relation contained at least 2 mappings and 0 or 1 is-a relations in the other ontology. In each of these cases the justification contained at least one mapping that the domain expert validated to be wrong or related and the wrong is-a relations were repaired by removing these mappings (see Figure 6 column 'final

[5] So the domain experts changed their original validation based on the reasoning support provided by RepOSE.

alignment RepOSE', except last row). In some cases repairing one wrong is-a relation also repaired others (e.g. removing mapping *hepatic porphyria* ← *porphyrias*, repairs two wrong is-a relations in MeSH: *porphyrias* → *porhyrias hepatic* and *porphyrias* → *drug induced liver injury*).

For the 9 missing is-a relations in ToxOntology and the 5 missing is-a relations in MeSH, possible repairing actions (using Source and Target sets) were generated. For most of these missing is-a relations the Source and Target sets were small, although for some there were too many elements in the set to provide for good visualization. For all these missing is-a relations the repairing constituted of adding the missing is-a relations themselves (Figure 7 column 'RepOSE'). In all but three cases this is what RepOSE recommended based on external knowledge from WordNet and UMLS. In 3 cases the system recommended to add other is-a relations, that were not considered correct by the domain experts (and thus wrong or based on a different view of the domain in the external domain knowledge).

After this repairing, we detected one new CMI in MeSH. This was validated as a wrong is-a relation and resulted in the removal of one more mapping (see Figure 6 column 'final alignment RepOSE' last row).

Discussion. Generally, detecting defects in ontologies without the support of a dedicated system is cumbersome and unreliable. In the case outlined in this paper RepOSE clearly provided a necessary support. Further, visualization of the justifications of possible defects was very helpful to have at hand as well as a graphical display of the possible defects within their contexts in the ontologies addressed. Moreover, RepOSE stored information about all changes made and their consequences as well as the remaining defects needing amendment.

As the set of CMIs was relatively small, it was possible for domain experts to perform a manual repair. They could focus on the pieces of ToxOntology that were related to the missing and wrong is-a relations. This allowed us to compare results of manual repair with those of repairment using RepOSE.

Regarding the changes in the alignment, for 11 term pairs the mapping was removed or changed in both approaches. For 2 term pairs the manual approach changed an is-a relation into an equivalence and for 2 other term pairs an is-a relation was changed into a wrong relation. These changes were not logically derivable and could not be found by RepOSE. For 3 of these term pairs the change came after the domain experts realized (using the justifications of the CMIs) that *metabolism* in MeSH has a different meaning than *metabolism* in ToxOntology. For 1 term pair (one but last row in Figure 6) the equivalence mapping was changed into wrong by the domain experts, while using RepOSE it was changed into an is-a relation. In the final alignment the RepOSE result was used. Further, through a second round of detection, using RepOSE an additional wrong mapping was detected and repaired, which was not found in the manual approach.

Regarding the addition of is-a relations to ToxOntology, the domain experts added one more is-a relation in the manual approach than in the approach using RepOSE. It could not be logically derived that *asthma* → *respiratory toxicity* was missing, but it was added by the domain experts in analogy to the repairing of another missing is-a relation.

In some cases, when using RepOSE, the justification for a missing is-a relation was removed after a wrong is-a relation was repaired by removing a mapping. For instance, after removing *metabolism (ToxicOntology)* $\leftarrow$ *metabolism (MeSH)*, there was no more justification for the missing is-a relation *hydrolysis* $\rightarrow$ *metabolism*. However, an advantage of RepOSE is that once a relation is validated as missing, RepOSE requires that it will be repaired and thus, this knowledge will be added, even without a justification.

Another advantage of RepOSE is that, for repairing a wrong is-a relation, it allows to remove multiple is-a relations and mappings in the justification, even though it may be sufficient to remove one. This was used, for instance, in the repair of the wrong is-a relation *phototoxicity* $\rightarrow$ *photosensitisation* in ToxOntology where *photosensitisation* $\equiv$ *photosensitivity disorders* and *phototoxicity* $\equiv$ *dermatitis phototoxic* were removed. Further, the repairing of one defect can lead to other defects being repaired. For instance, the removal of these two mappings also repaired the wrong is-a relation *photosensitivity disorders* $\rightarrow$ *dermatitis phototoxic* in MeSH. In general, RepOSE facilitates the computation and understanding of the consequences of repairing actions.

Interestingly, in this use case only mappings were removed to repair wrong is-a relations. This indicates that the ontology developers modeled the is-a structure decently. This kind of repair is not, however, a consistent outcome. For instance, in the experiment outlined in [11] involving debugging two ontologies and their alignment from the Anatomy track in OAEI 2010 (Adult Mouse Anatomy Dictionary (AMA) and the NCI Thesaurus anatomy (NCI-A), 14 is-a relations were removed from AMA and 11 from NCI-A, as well as 5 mappings. Further, in this use case all missing is-a relations were repaired by adding the missing is-a relations themselves. In the experiment in [11] in 27 cases in AMA and 11 cases in NCI-A a missing is-a relation was repaired using a more informative repairing action, thereby adding new knowledge that was not derivable from the ontologies and their alignment.

An identified constraint of RepOSE pertains to the fact that adding and removing is-a relations and mappings not appearing in the computations in RepOSE can be a demanding undertaking. Currently, these changes need to be conducted in the ontology files, but it would be useful to allow a user to do this via the system. For instance, it would have been useful to add *asthma* $\rightarrow$ *respiratory toxicity* via RepOSE.

3.3 Debugging using non-validated alignment

In the previous subsection the validated alignment was used as input. As a domain expert validated the mappings, they could be considered of high quality, although we showed that defects in the mappings were detected. In this subsection we perform an experiment with a non-validated alignment; we use the 41 mapping suggestions with a similarity value higher than or equal to 0.8 and use them initially as equivalence mappings.[6]

Using RepOSE (in 2 iterations) 16 non-redundant CMIs (27 in total), were computed for ToxOntology of which 6 were also computed in the debugging session described in 3.2. For MeSH 6 non-redundant CMIs (10 in total) were computed of which

[6] From the validation we know that these actually contain 29 equivalence mappings, 2 is-a mappings between a ToxOntology term and a MeSH term, 2 is-a mappings between a MeSH term and a ToxOntology term, 1 related term mapping and 7 wrong mappings.

2 were also computed earlier. As expected, the newly computed CMIs were all validated as wrong is-a relations and their computation was a result of wrong mappings. During the repairing 5 of the 7 wrong mappings were removed, and 2 initial mappings were changed into is-a mappings. RepOSE can thus be helpful in the validation of non-validated alignments - a domain expert will be able to detect and remove wrong mappings that lead to the logical derivation of wrong is-a relations, but wrong mappings that do not lead to logical derivation of wrong is-a relations, may not be found.

4 Conclusion

In this paper we presented an experience report on the debugging of ToxOntology, MeSH and an alignment. We showed the usefulness of RepOSE in detecting and repairing the structure of the ontologies and the alignment.

RepOSE is a logic-based debugging system[7] and detects defects based on logically derivable missing or wrong structure and mappings. In the future, we will investigate the integration of other detection approaches into RepOSE. Also, we will facilitate the adding and removing is-a relations and mappings that do not occur in the computation of the system. Finally, we will investigate the integration of RepOSE with SAMBO.

Acknowledgements. We thank the Swedish Research Council (Vetenskapsrådet), the Swedish e-Science Research Centre (SeRC) and the Swedish Civil Contingencies Agency for financial support.

References

1. DJ Dix, KA Houck, MT Martin, AM Richard, RW Setzer, and RJ Kavlock. The ToxCast program for prioritizing toxicity testing of environmental chemicals. *Toxicol. Sci.*, 95:5–12, 2007.
2. A Edberg, D Soeria-Atmadja, J Bergman Laurila, F Johansson, MG Gustafsson, and U Hammerling. Assessing relative bioactivity of chemical substances using quantitative molecular network topology analysis. *J. Chem. Inf. Model.*, 52:1238–1249, 2012.
3. J Euzenat and P Shvaiko. *Ontology Matching*. Springer, 2007.
4. B Hardy, N Douglas, C Helma, M Rautenberg, N Jeliazkova, V Jeliazkov, I Nikolova, R Benigni, O Tcheremenskaia, S Kramer, T Girschick, F Buchwald, J Wicker, A Karwath, M Gutlein, A Maunz, HS Sarimveis, G Melagraki, A Afantitis, P Sopasakis, D Gallagher, V Poroikov, D Filimonov, A Zakharov, A Lagunin, T Gloriozova, S Novikov, N Skvortsova, D Druzhilovsky, S Chawla, I Ghosh, S Ray, H Patel, and S Escher. Collaborative development of predictive toxicology applications. *J. Cheminform.*, 2:7, 2010.
5. Medical Subject Headings. http://www.nlm.nih.gov/mesh/.
6. N Jeliazkova and V Jeliazkova. AMBIT RESTful web services: an implementation of the OpenTox application programming interface. *J. Cheminform.*, 3:18, 2011.
7. E Jimenez-Ruiz, B Cuenca Grau, I Horrocks, and R Berlanga. Ontology integration using mappings: Towards getting the right logical consequences. In *Proceedings of the 6th European Semantic Web Conference*, volume 5554 of *LNCS*, pages 173–187. Springer, 2009.

[7] For an overview of current approaches to debugging we refer to [11].

8. RS Judson, RJ Kavlock, RW Setzer, EA Cohen Hubal, MT Martin, TB Knudsen, KA Houck, RS Thomas, BA Wetmore, and DJ Dix. Estimating toxicity-related biological pathway altering doses for high-throughput chemical risk assessment. *Chem. Res. Toxicol.*, 24:451–462, 2011.
9. RS Judson, AM Richard, DJ Dix, K Houck, F Elloumi, M Martin, T Cathey, TR Transue, R Spencer, and M Wolf. ACToR - aggregated computational toxicology resource. *Toxicol. Appl. Pharmacol.*, 233(1):7–13, 2008.
10. RS Judson, AM Richard, DJ Dix, K Houck, M Martin, R Kavlock, V Dellarco, T Henry, T Holderman, P Sayre, S Tan, T Carpenter, and E Smith. The toxicity data landscape for environmental chemicals. *Environ. Health Perspect.*, 117:685–695, 2009.
11. P Lambrix and V Ivanova. A unified approach for debugging is-a structure and mappings in networked taxonomies. *submitted*, 2012.
12. P Lambrix and Q Liu. Using partial reference alignments to align ontologies. In *Proceedings of the 6th European Semantic Web Conference*, volume 5554 of *LNCS*, pages 188–202, 2009.
13. P Lambrix, Q Liu, and H Tan. Repairing the missing is-a structure of ontologies. In *Proceedings of the 4th Asian Semantic Web Conference*, volume 5926 of *LNCS*, pages 76–90, 2009.
14. P Lambrix and H Tan. SAMBO - a system for aligning and merging biomedical ontologies. *Journal of Web Semantics*, 4(3):196–206, 2006.
15. P Lambrix and H Tan. A tool for evaluating ontology alignment strategies. *Journal on Data Semantics*, VIII:182–202, 2007.
16. W Lilienblum, W Dekant, H Foth, T Gebel, JG Hengstler, R Kahl, PJ Kramer, H Schweinfurth, and KM Wollin. Alternative methods to safety studies in experimental animals: role in the risk assessment of chemicals under the new European Chemicals Legislation (REACH). *Arch. Toxicol.*, 82:211–236, 2008.
17. F Maddah, D Soeria-Atmadja, P Malm, MG Gustafsson, and U Hammerling. Interrogating health-related public databases from a food toxicology perspective: Computational analysis of scoring data. *Food Chem Toxicol.*, 49:2830–2840, 2011.
18. CJ Mattingly, MC Rosenstein, AP Davis, GT Colby, JN Forrest Jr, and JL Boyer. The comparative toxicogenomics database: a cross-species resource for building chemical-gene interaction networks. *Toxicol. Sci.*, 92:587–595, 2006.
19. PubMed. http://www.ncbi.nlm.nih.gov/pubmed/.
20. AM Richard, LS Gold, and MC Nicklaus. Chemical structure indexing of toxicity data on the internet: moving toward a flat world. *Curr. Opin. Drug Discov. Devel.*, 9:314–325, 2006.
21. AM Richard, C Yang, and RS Judson. Toxicity data informatics: supporting a new paradigm for toxicity prediction. *Toxicol. Mech. Methods*, 18:103–118, 2008.
22. SJ Shukla, R Huang, CP Austin, and M Xia. The future of toxicity testing: a focus on in vitro methods using a quantitative high-throughput screening platform. *Drug Discov. Today*, 15:997–1007, 2010.
23. Unified Medical Language System. http://www.nlm.nih.gov/research/umls/.
24. H Tan, V Jakoniene, P Lambrix, J Aberg, and N Shahmehri. Alignment of biomedical ontologies using life science literature. In *Proceedings of the International Workshop on Knowledge Discovery in Life Science Literature*, volume 3886 of *LNBI*, pages 1–17, 2006.
25. O Tcheremenskaia, R Benigni, I Nikolova, N Jeliazkova, SE Escher ans M Batke, T Baier, V Poroikov, A Lagunin, M Rautenberg, and B Hardy. OpenTox predictive toxicology framework: toxicological ontology and semantic media wiki-based OpenToxipedia. *J. Biomed. Semantics*, 3 Suppl 1:S7, 2012.
26. GM Woodall and RB Goldberg. Summary of the workshop on the power of aggregated toxicity data. *Toxicol. Appl. Pharmacol.*, 233:71–75, 2008.
27. WordNet. http://wordnet.princeton.edu/.

A System for Debugging Taxonomies and their Alignments

Valentina Ivanova and Patrick Lambrix

Department of Computer and Information Science
and Swedish e-Science Researche Centre
Linköping University, SE-581 83 Linköping, Sweden

Abstract. Neither developing ontologies nor aligning ontologies are easy tasks, and often the resulting ontologies and alignments are not consistent or complete. Such ontologies and alignments, although often useful, also lead to problems when used in semantically-enabled applications. In this paper we briefly introduce a system that supports domain experts in detecting and repairing wrong and missing is-a relations and mappings.

1 Introduction

Neither developing nor aligning ontologies are easy tasks, and often the resulting ontologies and alignments are not consistent or complete. Such ontologies and alignments, although often useful, also lead to problems when used in semantically-enabled applications. Wrong conclusions may be derived or valid conclusions may be missed.

RepOSE (*R*epair of *O*ntological *S*tructure *E*nvironment) tackles the problem of debugging the is-a structure of a fundamental kind of ontologies, i.e., taxonomies, as well as the debugging of the mappings between taxonomies.

In this demonstration paper we briefly introduce RepOSE[1] (Section 2) and some experiments and projects in which RepOSE was used (Section3). However, for the theoretical background, algorithms, more detailed descriptions and related work we refer to [3]. For more detailed descriptions for the first and second cases in Section 3, we refer to [3, 2]. In Section 4 we introduce the demonstration at the First International Workshop on Debugging Ontologies and Ontology Mappings.

2 System

The input to RepOSE is an ontology network consisting of taxonomies and alignments between the taxonomies. The debugging process consists of the phases of detecting and validating possible defects, and repairing wrong and missing is-a relations and mappings. At any time during the process, the user can switch between different ontologies and alignments, start earlier phases, or switch between the repairing of wrong and missing is-a relations and mappings. For each of the steps in the debugging process, RepOSE

[1] The version of RepOSE described in this paper is an extension of earlier described versions. Previous versions dealt only with missing and/or wrong is-a relations, but not with mappings.

can recommend possible actions. The process ends when there are no more defects or defect suggestions to deal with.

In the current version of RepOSE we have focused on detecting defects using the knowledge inherent in the network. RepOSE suggests defects in the form of candidate missing is-a relations and mappings. Candidate missing is-a relations in an ontology are is-a relations that can be derived from the network but not from the ontology alone. Candidate missing mappings between two ontologies are mappings that can be derived from the network but not from the ontologies and their alignment alone. These candidate missing is-a relations and mappings are then validated by a domain expert and classified as missing and wrong is-a relations and mappings (Figure 1).

For these defects RepOSE computes repairing actions, i.e., is-a relations or mappings to add to and remove from the ontologies and the alignments such that (i) the missing is-a relations will be derivable from their host ontologies (ii) the missing mappings will be derivable from the host ontologies of the concepts in the mappings, and their alignment, and (iii) the wrong is-a relations and mappings will not be derivable from the ontology network. For wrong is-a relations and mappings RepOSE shows their justification and the domain expert can select is-a relations and mappings to remove (Figure 2)[2]. For missing is-a relations, RepOSE shows two panels, where it is guaranteed that when an is-a relation or mapping is added between an element in the first panel and an element in the second panel, the missing is-a relation or mapping will be repaired (Figure 3). Upon repairing, RepOSE computes all the consequences of the repair.

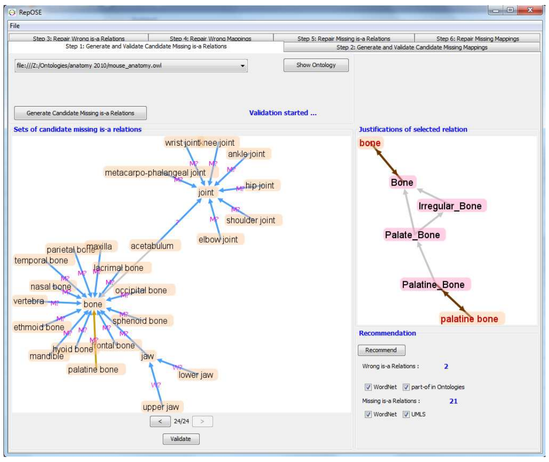
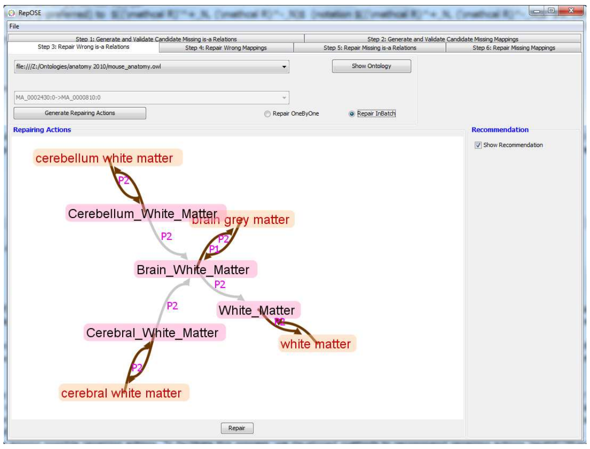

RepOSE
File
Step 3: Repair Wrong is-a Relations
Step 4: Repair Wrong Mappings
Step 5: Repair Missing is-a Relations
Step 6: Repair Missing Mappings
Step 1: Generate and Validate Candidate Missing is-a Relations
Step 2: Generate and Validate Candidate Missing Mappings
file:///Z:/Ontologies/anatomy 2010/mouse_anatomy.owl
Show Ontology
Generate Candidate Missing is-a Relations
Validation started ...
Sets of candidate missing is-a relations
wrist joint
knee joint
ankle joint
metacarpo-phalangeal joint
joint
hip joint
shoulder joint
elbow joint
acetabulum
parietal bone
maxilla
temporal bone
lacrimal bone
nasal bone
occipital bone
vertebra
bone
sphenoid bone
ethmoid bone
hyoid bone
frontal bone
mandible
jaw
lower jaw
palatine bone
upper jaw
24/24
Validate
Justifications of selected relation
bone
Bone
Irregular_Bone
Palate_Bone
Palatine_Bone
palatine bone
Recommendation
Recommend
Wrong is-a Relations : 2
WordNet
part-of in Ontologies
Missing is-a Relations : 21
WordNet
UMLS

Fig. 1. Generating and validating candidate missing is-a relations and mappings.

[2] The screenshots in the figures are for tabs related to is-a relations. Similar tabs exist related to mappings.


RepOSE
File
Step 1: Generate and Validate Candidate Missing is-a Relations
Step 2: Generate and Validate Candidate Missing Mappings
Step 3: Repair Wrong is-a Relations
Step 4: Repair Wrong Mappings
Step 5: Repair Missing is-a Relations
Step 6: Repair Missing Mappings
Show Ontology
Generate Repairing Actions
Repairing Actions
Recommendation
cerebellum white matter
P2
Cerebellum_White_Matter
brain grey matter
P2
P2
P1
Brain_White_Matter
P2
P2
White_Matter
P2
Cerebral_White_Matter
white matter
P2
cerebral white matter
Repair

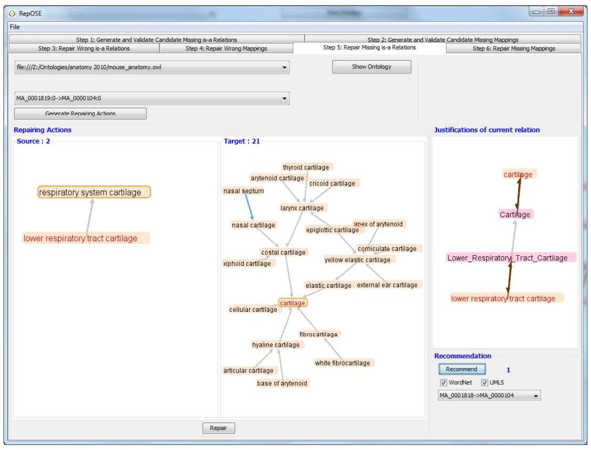
file:///Z:/Ontologies/anatomy 2010/mouse_anatomy.owl

MA_0002430:0->MA_0000810:0

Repair OneByOne [✓ Repair InBatch]

[✓] Show Recommendation

Fig. 2. Repairing wrong is-a relations.



file:///Z:/Ontologies/anatomy 2010/mouse_anatomy.owl

MA_0001819:0->MA_0000104:0

[✓] WordNet [✓] UMLS

MA_0001818->MA_0000104

Fig. 3. Repairing missing is-a relations.

3 Uses of RepOSE

The current version of RepOSE has been used in a number of cases. The first case is work that was performed for the Swedish National Food Agency. The input data contained a large ontology, a small ontology and an alignment. The existing structure was of good quality. Most defects related to missing is-a relations and wrong mappings. The second case uses the ontologies and alignment of a track in the Ontology Alignment Evaluation Initiative. The input data contained two larger ontologies and an alignment. Defects in the structures of the ontologies and the alignment were repaired. In this case using our approach also new knowledge was added to the network. For the third case the input data contained five smaller ontologies and four alignments. In this case also missing mappings could be found and new alignments were generated.

ToxOntology - MeSH. RepOSE has been used to debug ToxOntology and an alignment to MeSH [4]. ToxOntology is a toxicology ontology and was created within an informatics system development at the Swedish National Food Agency as part of an initiative to facilitate the identification of adequate substance-associated health effects. ToxOntology is an OWL2 ontology, encompassing 263 concepts and 266 asserted is-a relations. Further, an alignment with MeSH was desired to obtain an indirect index to the scientific literature. MeSH is a thesaurus of the National Library of Medicine. As MeSH contains many concepts not related to the domain of toxicology, a part of MeSH was used. This part contained 9,878 concepts and 15,786 asserted is-a relations.

In the initial detection phase RepOSE generated 12 non-redundant candidate missing is-a relations for ToxOntology (34 in total) of which 9 were validated by the domain experts as missing and 3 as wrong. For MeSH, RepOSE generated 17 non-redundant candidate missing is-a relations (among which 2 relations represented one equivalence relation - 32 candidate missing is-a relations in total) of which 5 were validated as missing and the rest as wrong. For the 3 wrong is-a relations for ToxOntology and the 12 wrong is-a relations for MeSH, the justifications contained at least one mapping that the domain expert validated to be wrong or related and the wrong is-a relations were repaired by removing these mappings. The 9 missing is-a relations in ToxOntology and the 5 missing is-a relations in MeSH were repaired by adding the missing is-a relations themselves. In all but three cases this was what RepOSE recommended based on external knowledge from WordNet and UMLS. After this repairing, one new candidate missing is-a relation was detected in MeSH, which was validated as a wrong is-a relation and resulted in the removal of one more mapping.

OAEI 2010 Anatomy. RepOSE was also used during an experiment on a network consisting of the two ontologies and the alignment from the Anatomy track in the Ontology Alignment Evaluation Initiative (OAEI, [6]) 2010. The Adult Mouse Anatomy Dictionary (AMA, [1]) contains 2744 concepts and 1807 asserted is-a relations, while the NCI Thesaurus anatomy (NCI-A, [5]) contains 3304 concepts and 3761 asserted is-a relations. The alignment contains 986 equivalence and 1 subsumption mapping between AMA and NCI-A. These ontologies as well as the alignment were developed by domain experts. The experiment was performed by a domain expert.

The system detected 200 candidate missing is-a relations in AMA of which 123 were non-redundant. Of these non-redundant candidate missing is-a relations 102 were validated to be missing is-a relations and 21 were validated to be wrong is-a relations. For NCI-A 127 candidate missing is-a relations of which 80 were non-redundant, were detected. Of these non-redundant candidate missing is-a relations 61 were validated to be missing is-a relations and 19 were validated to be wrong is-a relations. To repair these defects 85 is-a relations were added to AMA and 57 to NCI-A, 13 is-a relations were removed from AMA and 12 from NCI-A, and 12 mappings were removed from the alignment. In 22 cases in AMA and 8 cases in NCI-A a missing is-a relation was repaired using a more informative repairing action (not derivable from the network), thereby adding new knowledge to the network.

The recommendations seemed useful. Regarding candidate missing is-a relations, 81 and 27 recommendations that the relation should be validated as a missing is-a relation, were accepted for AMA, respectively NCI-A, while 8 and 2 were rejected. When the system recommended that a candidate missing is-a relation should be validated as a wrong is-a relation, the recommendation was accepted in 7 out of 20 cases for AMA and 6 out of 8 cases for NCI-A. The recommendations regarding repairing missing is-a relations were accepted in 69 out of 85 cases for AMA and 43 out of 57 cases for NCI-A.

OAEI 2010 Bibliography. Another experiment is based on the bibliography case in the OAEI 2010. This case consists of 5 smaller ontologies in the bibiography domain (called 101, 301, 302, 303 and 304 in the OAEI set), with between 13 and 56 concepts each. The ontologies are connected in a star shape. (Ontology 101 has 22, 23, 18 and 30 mappings to 301, 302, 303 and 304, respectively. There are no mappings among the other pairs of ontologies.) Initially, RepOSE found 6 candidate missing is-a relations in ontology 101, 5 in ontology 304 and 1 in each of the other ontologies. During the repairing 2 additional candidate missing is-a relations were found for ontology 101, 1 for 302 and 3 for 304. Of all these 14 were validated as missing and 5 as wrong. Further, for the pairs of ontologies for which no alignment existed, candidate missing mappings were generated of which 187 were validated as missing and 15 as wrong. These defects were repaired by adding 19 is-a relations in ontologies and 181 mappings, and removing 7 is-a relations in ontologies and 10 mappings. The missing is-a relations and mappings were in 18 cases repaired by more informative (with respect to the network) repairing actions. We note that using RepOSE alignments were generated for each pair of ontologies for which no alignment existed previously.

4 Demonstration

In the demonstration we guide the visitors through a debugging session using (parts of) the ontologies from OAEI 2010 Anatomy, as well as a debugging session for OAEI 2010

Bibliography. Further, we explain the algorithms for the detection and validation of defects, as well as the generation, recommendation and execution of repairing actions.

Acknowledgements. We thank the Swedish Research Council (Vetenskapsrådet) and the Swedish e-Science Research Centre (SeRC) for financial support.

References

1. AMA. Adult mouse anatomical dictionary. http://www.informatics.jax.org/searches/AMA_form.shtml.
2. V Ivanova, J Laurila Bergman, U Hammerling, and P Lambrix. Debugging taxonomies and their alignments: the ToxOntology - MeSH use case. *1st International Workshop on Debugging Ontologies and Ontology Mappings*, 2012.
3. P Lambrix and V Ivanova. A unified approach for debugging is-a structure and mappings in networked taxonomies. *Report*, 2012.
4. MeSH. Medical subject headings. http://www.nlm.nih.gov/mesh/.
5. NCI-A. National cancer institute - anatomy. http://www.cancer.gov/cancerinfo/terminologyresources/.
6. OAEI. Ontology alignment evaluation initiative. http://oaei.ontologymatching.org/.