Planning Accessible Explanations for Entailments in OWL Ontologies

Tu Anh T. Nguyen, Richard Power, Paul Piwek, Sandra Williams
The Open University
Milton Keynes, United Kingdom
{t.nguyen,r.power,p.piwek,s.h.williams}@open.ac.uk

Abstract

A useful enhancement of an NLG system for verbalising ontologies would be a module capable of explaining undesired entailments of the axioms encoded by the developer. This task raises interesting issues of content planning. One approach, useful as a baseline, is simply to list the subset of axioms relevant to inferring the entailment; however, in many cases it will still not be obvious, even to OWL experts, why the entailment follows. We suggest an approach in which further statements are added in order to construct a proof tree, with every step based on a relatively simple deduction rule of known difficulty; we also describe an empirical study through which the difficulty of these simple deduction patterns has been measured.

1 Introduction

A practical problem in developing ontologies for the semantic web is that mistakes are hard to spot. One reason for this lies in the opacity of the standard OWL formalisms, such as OWL/RDF, which are designed for efficient processing by computer programs and not for fast comprehension by people. Various tools have been proposed to address this problem, including not only graphical interfaces such as Protégé, but NLG (Natural Language Generation) programs that *verbalise* the axioms of an ontology as text (Kaljurand and Fuchs, 2007; Schwitter and Meyer, 2007; Hart et al., 2008). Using such a tool, a mistaken axiom presented through a sentence like 'Every person is a movie' immediately leaps to the eye.

Although there is evidence that verbalisation helps developers to check individual axioms (Stevens et al., 2011), there remains a more subtle problem of undesired *entailments*, often based on interactions among axioms. The difference between axioms and entailments is that whereas axioms are statements encoded by the developer, entailments are statements inferred from axioms by automated reasoners such as FaCT++ (Tsarkov and Horrocks, 2006). Because reasoning systems interpret statements absolutely literally, it is quite common for apparently innocuous axioms to lead to absurd conclusions such as 'Everything is a person', 'Nothing is a person', or indeed 'Every person is a movie'. The standard reasoning algorithms, based on tableau algorithms, will compute these entailments efficiently, but they provide no information that helps explain *why* an undesired conclusion was drawn, and hence which axiom or axioms need to be corrected.

To provide an explanation of an entailment, the first step is obviously to determine which axioms are relevant to the inference. A set of relevant axioms is known technically as a *justification* of the entailment, defined as any minimal subset of the ontology from which the entailment can be drawn (Kalyanpur, 2006). The minimality requirement here means that if any axiom is removed from a justification, the entailment will no longer be inferable.

Drawing on Kalyanpur's work, the most direct strategy for planning an explanation is simply to verbalise the axioms in the justification, followed by the entailment, with no additional content. This strategy serves as a useful baseline for comparison, and might even be effective for some simple justi-

Entailment	*Person* $\sqsubseteq$ *Movie*	*Every person is a movie.*
Justification	1. *GoodMovie* $\equiv \forall$*hasRating.FourStars* 2. *Domain(hasRating)* = *Movie* 3. *GoodMovie* $\sqsubseteq$ *StarRatedMovie* 4. *StarRatedMovie* $\sqsubseteq$ *Movie*	1. *A good movie is anything that only has ratings of four stars.* 2. *Anything that has a rating is a movie.* 3. *Every good movie is a star-rated movie.* 4. *Every star-rated movie is a movie.*

Table 1: An example justification that requires further explanation

fications; 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 explanation (Horridge et al., 2009). This raises two problems of content planning that we now address: (a) how we can ascertain that further explanation is needed, and (b) what form such explanation should take.

2 Explaining complex justifications

An example of a justification requiring further explanation is shown in Table 1. Statements are presented in mathematical notation in the middle column (rather than in OWL, which would take up a lot more space), with a natural language gloss in the right column. Since these sentences are handcrafted they should be more fluent than the output of a verbaliser, but even with this benefit, it is extremely hard to see why the entailment follows.

The key to understanding this inference lies in the first axiom, which 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*. From this it follows that people, parrots, parsnips, or in general things that cannot have a rating, all belong to the second class, which is asserted to be equivalent to the class of good movies. If individuals with no rating are good movies, then by axioms 3 and 4 they are also movies, so we are left with two paradoxical statements: individuals *with* a rating are movies (axiom 2), and individuals *without* a rating are movies (the intermediate conclusion just derived). Since everything that exists must either have some rating or no rating, we are driven to the conclusion that everything is a movie, from which it follows that any person (or parrot, etc.) must also be a movie: hence the entailment. Our target explanation for this case is as follows:

> Every person is a movie because the ontology implies that everything is a movie.
> Everything is a movie because (a) anything that has a rating is a movie, and (b) anything that has no rating at all is a movie.
> Statement (a) is stated in axiom 2 in the justification. Statement (b) is inferred because the ontology implies that (c) anything that has no rating at all is a good movie, and (d) every good movie is a movie.
> Statement (d) is inferred from axioms 3 and 4 in the justification. 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.

Note that in this or any other intelligible explanation, a path is traced from premises to conclusion by introducing a number of intermediate statements, or *lemmas*. Sometimes a lemma merely unpacks part of the meaning of an axiom — the part that actually contributes to the entailment. This is clearly what we are doing when we draw from axiom 1 the implication that all individuals with no ratings are good movies. Alternatively a lemma could be obtained by combining two axioms, or perhaps even more. By introducing appropriate lemmas of either type, we can construct a *proof tree* in which the root node is the entailment, the terminal nodes are the axioms in the justification, and the other nodes are lemmas. An explanation based on a proof tree should be easier to understand because it replaces a single complex inference step with a number of simpler ones.

Assuming that some kind of proof tree is needed, the next question is how to construct proof trees that provide *effective* explanations. Here two conditions need to be met: (1) the proof tree should be correct, in the sense that all steps are valid; (2) it should be accessible, in the sense that all steps are understandable. As can be seen, one of these conditions is logical, the other psychological. Several research groups have proposed methods for producing logically correct proof trees for description logic (McGuinness, 1996; Borgida et al., 1999; Horridge et al., 2010), but explanations planned in this way will not necessarily meet our second requirement. In fact they could fail in two ways: either they might employ a single reasoning step that most people cannot follow, or they might unduly complicate the text by including multiple steps where a single step would have been understood equally well. We believe this problem can be addressed by constructing the proof tree from deduction rules for which the intuitive difficulty has been *measured* in an empirical study.[1]

3 Collecting Deduction Rules

For our purposes, a deduction rule consists of a conclusion (i.e., an entailment) and up to three premises from which the conclusion logically follows. Both conclusion and premises are generalised by using variables that abstract over class and property names, as shown in Table 2, where for example the second rule corresponds to the well-known syllogism that from 'Every A is a B' and 'Every B is a C', we may infer 'Every A is a C'.

Our deduction rules were derived through a corpus study of around 500 OWL ontologies. First we computed entailment-justification pairs using the method described in Nguyen et al. (2010), and collated them to obtain a list of deduction patterns ranked by frequency. From this list, we selected patterns that were simple (in a sense that will be explained shortly) and frequent, subsequently adding 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 algorithm.[2] The deduction rules required for the previous example are shown in Table 2. So far, 41 deduction rules have been obtained in this way; these are sufficient to generate proof trees for 48% of the justifications of subsumption entailments in the corpus (i.e., over 30,000 justifications).

As a criterion of *simplicity* we considered the number of premises (we stipulated not more than three) and also what is called the 'laconic' property (Horridge et al., 2008) — 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 *understandable* by people; we return to this issue in section 5, which describes an empirical test of the understandability of the rules.

4 Constructing Proof Trees

A proof tree can be defined as any tree linking the axioms of a justification (terminal nodes) to an entailment (root node), in such a way that every local tree (i.e., every node and its children) corresponds to a deduction rule. This means that if the entailment and justification already correspond to a deduction rule, no further nodes (i.e., lemmas) need to be added. Otherwise, a proof can be sought by applying the deduction rules, where possible, to the terminal nodes, so introducing lemmas and growing the tree bottom-up towards the root. Exhaustive search using this method may yield zero, one or multiple solutions — e.g., for our example two proof trees were generated, as depicted in Figure 1.[3]

5 Measuring understandability

To investigate the difficulty of deduction rules empirically, we have conducted a survey in which 43 participants (mostly university staff and students unfamiliar with OWL) were shown the premises of the rule, expressed as English sentences concerning fictitious entities, and asked to choose the correct conclusion from four alternatives. They were also asked to rate the difficulty of this choice on a five-point scale. For instance, in one problem the premises

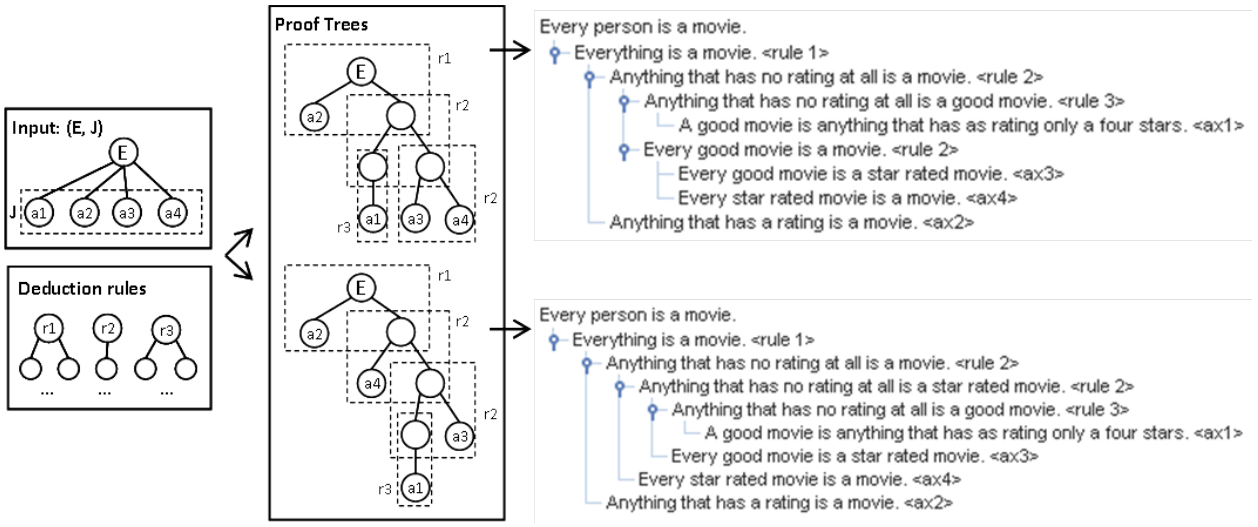
[1]Deduction rules were previously used by Huang for reconstructing machine-generated mathematical proofs; however, these rules were not for description logic based proofs and assumed to be intuitive to people (Huang, 1994). The output proofs were then enhanced (Horacek, 1999) and verbalised (Huang, 1994).

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

[3]In the current implementation, the proof tree can also be developed by adding lemmas that unpack part of the meaning of an axiom, using the method proposed by Horridge et al.(2008). These steps in the proof are not always obvious, so their understandability should also be measured.

ID	Deduction Rule	Example	Success Rate
1	$\forall r.\bot \sqsubseteq C$ $\exists r.\top \sqsubseteq C$ $\rightarrow \top \sqsubseteq C$	*Anything that has no ratings at all is a movie.* *Anything that has a rating is a movie.* *→ Everything is a movie.*	65%
2	$C \sqsubseteq D$ $D \sqsubseteq E$ $\rightarrow C \sqsubseteq E$	*Anything that has no ratings at all is a good movie.* *Every good movie is a movie.* *→ Anything that has no ratings at all is a movie.*	88%
3	$C \equiv \forall r.D$ $\rightarrow \forall r.\bot \sqsubseteq C$	*A good movie is anything that only has ratings of four stars.* *→ Anything that has no ratings at all is a good movie.*	—

Table 2: Deduction rules for the example in Table 1



Every person is a movie.
Everything is a movie. <rule 1>
Anything that has no rating at all is a movie. <rule 2>
Anything that has no rating at all is a good movie. <rule 3>
A good movie is anything that has as rating only a four stars. <ax1>
Every good movie is a movie. <rule 2>
Every good movie is a star rated movie. <ax3>
Every star rated movie is a movie. <ax4>
Anything that has a rating is a movie. <ax2>

Every person is a movie.
Everything is a movie. <rule 1>
Anything that has no rating at all is a movie. <rule 2>
Anything that has no rating at all is a star rated movie. <rule 2>
Anything that has no rating at all is a good movie. <rule 3>
A good movie is anything that has as rating only a four stars. <ax1>
Every good movie is a star rated movie. <ax3>
Every star rated movie is a movie. <ax4>
Anything that has a rating is a movie. <ax2>

Figure 1: Proof trees generated by our current system

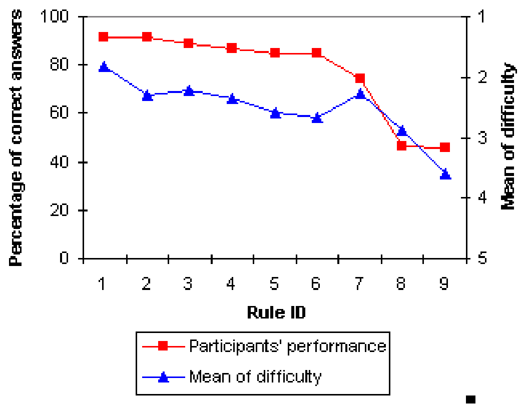


Figure 2: Results of the empirical study. In our difficulty scale, 1 means 'very easy' and 5 means 'very difficult'

were 'Every verbeeg is a giantkin; no giantkin is a verbeeg.'; to answer correctly, participants had to tick 'Nothing is a verbeeg' and not 'Nothing is a giantkin'.

So far 9/41 deduction rules have been measured in this way. Figure 2 shows the success rates and the means of difficulty of those rules. For most problems the success rates were around 80%, confirming that the rules were understandable, although in a few cases performance fell to around 50%, suggesting that further explanation would be needed. The study also indicates a statistically significant relationship between the accuracy of the participants' performance and their perceptions of difficulty (r = 0.82, p $<$ 0.01). Two of the three rules in Table 2 were measured in this way. The third rule has not been tested yet; however, its success rate is expected to be very low as it was proved to be a very difficult inference (Horridge et al., 2009).

6 Conclusion

This paper has reported our work in progress on content planning for explanations of entailments. The main steps involved in the planning process are sum-

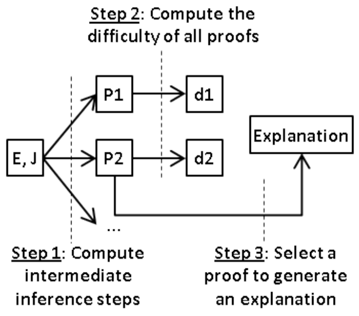

Step 2: Compute the difficulty of all proofs
E, J
P1
P2
...
d1
d2
Explanation
Step 1: Compute intermediate inference steps
Step 3: Select a proof to generate an explanation

Figure 3: Our approach for the content planning. E, J, Pn are entailments, justifications and proofs respectively; d1 and d2 are difficulty scores and d2 $\leq$ d1

marised in Figure 3. We have focused on one aspect: the introduction of lemmas that mediate between premises and conclusion, so organising the proof into manageable steps. Lemmas are derived by applying deduction rules collected through a corpus study on entailments and their justifications. Through a survey we have measured the difficulty of some of these rules, as evidenced by performance on the task of choosing the correct conclusion for given premises. These measures should indicate which steps in a proof are relatively hard, and thus perhaps in need of further elucidation, through special strategies that can be devised for each problematic rule. Our hypothesis is that these measures will also allow an accurate assessment of the difficulty of a candidate proof tree, so providing a criterion for choosing among alternatives — e.g., by using the success rates as an index of difficulty, we can sum the index over a proof tree to obtain a simple measure of its difficulty. Our verbaliser currently translates OWL statements literally, and needs to be improved to make sure any verbalisations do not give rise to unwanted presuppositions and Gricean implicatures.

Acknowledgments

This research was undertaken as part of the ongoing SWAT project (Semantic Web Authoring Tool), which is supported by the UK Engineering and Physical Sciences Research Council (EPSRC). We thank our colleagues and the anonymous viewers.

References

Alexander Borgida, Enrico Franconi, Ian Horrocks, Deborah L. McGuinness, and Peter F. Patel-Schneider. 1999. Explaining $\mathcal{ALC}$ Subsumption. In *DL 1999, International Workshop on Description Logics*.

Glen Hart, Martina Johnson, and Catherine Dolbear. 2008. Rabbit: developing a control natural language for authoring ontologies. In *ESWC 2008, European Semantic Web Conference*, pages 348–360.

Helmut Horacek. 1999. Presenting Proofs in a Human-Oriented Way. In *CADE 1999, International Conference on Automated Deduction*, pages 142–156.

Matthew Horridge, Bijan Parsia, and Ulrike Sattler. 2008. Laconic and Precise Justifications in OWL. In *ISWC 2008, International Semantic Web Conference*, pages 323–338.

Matthew Horridge, Bijan Parsia, and Ulrike Sattler. 2009. Lemmas for Justifications in OWL. In *DL 2009, International Workshop on Description Logics*.

Matthew Horridge, Bijan Parsia, and Ulrike Sattler. 2010. Justification Oriented Proofs in OWL. In *ISWC 2010, International Semantic Web Conference*, pages 354–369.

Xiaorong Huang. 1994. *Human Oriented Proof Presentation: A Reconstructive Approach.* Ph.D. thesis, The University of Saarbrücken, Germany.

Kaarel Kaljurand and Norbert Fuchs. 2007. Verbalizing OWL in Attempto Controlled English. In *OWLED 2007, International Workshop on OWL: Experiences and Directions*.

Aditya Kalyanpur. 2006. *Debugging and repair of OWL ontologies*. Ph.D. thesis, The University of Maryland, US.

Deborah Louise McGuinness. 1996. *Explaining reasoning in description logics*. Ph.D. thesis, The State University of New Jersey, US.

Tu Anh T. Nguyen, Paul Piwek, Richard Power, and Sandra Williams. 2010. Justification Patterns for OWL DL Ontologies. Technical Report TR2011/05, The Open University, UK.

Rolf Schwitter and Thomas Meyer. 2007. Sydney OWL Syntax - towards a Controlled Natural Language Syntax for OWL 1.1. In *OWLED 2007, International Workshop on OWL: Experiences and Directions*.

Robert Stevens, James Malone, Sandra Williams, Richard Power, and Allan Third. 2011. Automating generation of textual class definitions from OWL to English. *Journal of Biomedical Semantics*, 2(S 2:S5).

Dmitry Tsarkov and Ian Horrocks. 2006. FaCT++ Description Logic Reasoner: System Description. In *IJCAR 2006, International Joint Conference on Automated Reasoning*, pages 292–297.