From Justifications to Proofs for Entailments in OWL

Matthew Horridge[1], Bijan Parsia[1], Ulrike Sattler[1]

The University of Manchester, UK

1 Introduction

Over the past few years there has been a significant amount of interest in the area of explaining entailments in OWL ontologies. Without some kind of tool support, it can be very difficult, or even impossible, to work out why entailments arise in ontologies. Even in small ontologies that only contain tens of axioms, there can be multiple reasons for an entailment, none of which may be obvious. It is for this reason that there has recently been a lot of focus on generating explanations for entailments in ontologies. In the OWL world, *justifications* are a popular form of explanation for entailments. A justifications is a minimal subset of an ontology that is sufficient for an entailment to hold[1, 7, 10]. More precisely, for an ontology $\mathcal{O}$ and an entailment η where $\mathcal{O} \models \eta$ ($\mathcal{O}$ entails η), a set of axioms $\mathcal{J}$ is a justification for η with respect to $\mathcal{O}$ if $\mathcal{J} \subseteq \mathcal{O}$, $\mathcal{J} \models \eta$ and, for all $\mathcal{J}' \subsetneq \mathcal{J}$, $\mathcal{J}' \not\models \eta$. Additionally, $\mathcal{J}$ is simply a justification (without respect to $\mathcal{O}$) if $\mathcal{J} \models \eta$ and, if $\mathcal{J}' \subsetneq \mathcal{J}$, then $\mathcal{J}' \not\models \eta$.

Virtually all mainstream ontology editors such as Protégé-4, Swoop, and Top Braid Composer provide support for generating justifications as explanations for arbitrary entailments. Justifications have proved enormously useful for understanding and debugging ontologies. In [7], Kalyanpur presents a user study which showed that the availability of justifications had a significant positive impact on the ability of users to successfully diagnose and repair an ontology. Recently, justifications have been used for debugging very large ontologies such as SNOMED [2], where the size of the ontology prohibits efficient manual debugging.

Despite the utility of justifications, and the fact that the availability of a justification as a form of an explanation is nearly always better than nothing, in certain cases people can struggle to understand how a justification supports an entailment. In fact, in a recent study [5], it has been shown that people, who have expertise in building OWL ontologies, can find it very difficult and even impossible to understand some justifications.

The work presented in this paper looks at solving this problem using *Justification Oriented Proofs*. Given a justification, intermediate inference steps, called lemmas, are automatically derived to bridge the gap between the axioms in the justification and the entailment. A justification oriented proof shows in a stepwise way how the lemmas, and ultimately the entailment, follow from the justification. The framework makes use of a "complexity model", which predicts how easy or difficult it is for a user to understand a justification, and is used for selecting the lemmas to insert into a proof.

Proceedings of OWL: Experiences and Directions 2009 (OWLED 2009),
Rinke Hoekstra and Peter F. Patel-Schneider, editors. http://www.webont.org/owled/2009

2 Background and Motivation

As mentioned previously, people with varying levels of expertise in OWL can find justifications difficult or even impossible to understand. In order to give a flavour of why this might be the case, two example justifications are presented in Figures 1 and 2. The reader is encouraged to work through and understand these justifications in order to appreciate the issues.

Consider the justification shown in Figure 1, which is for the entailment $\mathsf{Person} \sqsubseteq \bot$ (i.e. Person is unsatisfiable). This is a justification for an entailment from a real ontology about movies that was posted to the Protégé mailing list. The person who posted it could not understand why Person was unsatisfiable, even when presented with the justification. When this justification was shown to a range of people, some of them having worked with OWL for several years, many of them struggled to understand it. In fact, some of the people claimed that it was not a justification for the entailment.

$$\mathsf{Person} \sqsubseteq \neg\mathsf{Movie}$$
$$\mathsf{RRated} \sqsubseteq \mathsf{CatMovie}$$
$$\mathsf{CatMovie} \sqsubseteq \mathsf{Movie}$$
$$\mathsf{RRated} \equiv (\exists\mathsf{hasScript.ThrillerScript}) \sqcup (\forall\mathsf{hasViolenceLevel.High})$$
$$\mathsf{Domain}(\mathsf{hasViolenceLevel},\ \mathsf{Movie})$$

Fig. 1. A justification for $\mathsf{Person} \sqsubseteq \bot$

Figure 2 presents an example justification for the entailment Newspaper(DailyMirror) (read as DailyMirror is an instance of Newspaper). When this justification was shown to people, many of them were put off by the number and variety of axioms in the justification. Those that weren't put off reported that they found it very difficult to work through.

In previous work [5], a user study was carried out where people were asked to view and understand justifications. Participants had varying levels of experience in OWL, ranging from less than six months to over four years. Some of the participants were bio-informaticians, while others worked on OWL tools such as editors and reasoners. In the study participants were shown a series of justifications and asked to rank them based on how difficult they found it to understand the justifications. The results of the study, confirm that there is indeed a problem, and that there are naturally occurring justifications that people find *very difficult* or even *impossible* to understand.

In essence, while a justification gathers together the axioms, or *premises*, for an entailment, it is left up to the person reading the justification to figure out how these premises interplay with each other to give rise to the entailment in question. In some cases, for example those presented in Figures 1 and 2, the

$\mathsf{InverseProperties}(\mathsf{hasPet},\ \mathsf{isPetOf})$
$\mathsf{isPetOf}(\mathsf{Rex}, \mathsf{Mick})$
$\mathsf{Domain}(\mathsf{hasPet},\ \mathsf{Person})$
$\mathsf{Male}(\mathsf{Mick})$
$\mathsf{reads}(\mathsf{Mick}, \mathsf{DailyMirror})$
$\mathsf{drives}(\mathsf{Mick}, \mathsf{Q123ABC})$
$\mathsf{Van}(\mathsf{Q123ABC})$
$\mathsf{Van} \sqsubseteq \mathsf{Vehicle}$
$\mathsf{WhiteThing}(\mathsf{Q123ABC})$
$\mathsf{Driver} \equiv \mathsf{Person} \sqcap \exists \mathsf{drives}.\mathsf{Vehicle}$
$\mathsf{Driver} \sqsubseteq \mathsf{Adult}$
$\mathsf{Man} \equiv \mathsf{Adult} \sqcap \mathsf{Male} \sqcap \mathsf{Person}$
$\mathsf{WhiteVanMan} \equiv \mathsf{Man} \sqcap \exists \mathsf{drives}.(\mathsf{Van} \sqcap \mathsf{WhiteThing})$
$\mathsf{WhiteVanMan} \sqsubseteq \forall \mathsf{reads}.\mathsf{Tabloid}$
$\mathsf{Tabloid} \sqsubseteq \mathsf{Newspaper}$

Fig. 2. A justification for $\mathsf{Newspaper}(\mathsf{DailyMirror})$

"gap" between seeing the premises and understanding how they give rise to the entailment is too large. So much so, that it is very difficult, or even impossible, for a person to understand the justification.

It is arguable that, what is needed, is some kind of "proof", that explicates the various *steps* involved in understanding how a justification supports an entailment. In what follows various approaches to a solution are discussed and *Justification Oriented Proofs* are presented as the chosen solution.

3 Proof Based Explanation Techniques

As stated previously in the Introduction, a justification for an entailment η may be regarded as the set of *premises* for some proof that shows how η follows from it. When searching for a solution to the problem of understanding justifications, it therefore seems fruitful to explore the space of presenting proofs to users.

Broadly speaking, proof based explanation techniques are typically inspired by Natural Deduction [4]. A Natural Deduction proof shows how a conclusion (a formula) follows from a set of premises (set of formulae) by presenting a series of intermediate formulae that step from the premises to the conclusion. Each formula in the proof is either a premise, the conclusion, or a formula which was *derived* from previous formulae in the proof via the application of syntactic transformation rules. A natural deduction proof begins with the premises (axioms in a justification), and ends with the conclusion (entailment). It is fre-

quently claimed that Natural Deduction mimics human reasoning, hence the name, and so is suitable for presenting proofs to humans.

In terms of Description Logics, there have been various efforts to provide frameworks for constructing explanation based proofs. One of the most notable frameworks was presented by Borgida et al. in "Explaining $\mathcal{ALC}$ Subsumption" [3]. In this work, a *baseline form of explanation* for an entailment is considered to be the proof obtained by extracting a completion tree, and the steps used to derive it, from a tableaux reasoner. They argue that due to the refutation based nature of, and normalising transformations, such as de Morgans rules, tableau proofs are *difficult* and *unnatural* for end users to examine and understand. They also state that it is undesirable to have an explanation component that is dissociated from the implementation of a reasoner (tableaux reasoner), citing reasons of efficiency and possible *deviation between implementation and explanation*. The authors therefore turn to Sequent Calculi, which are similar to Natural Deduction Calculi, for generating higher level proofs. The result is proofs that consist of a steps that *parallel "steps" in tableaux based proofs*, but that do not expose the refutation aspect of tableaux reasoning, and are chained together with "easily explainable" transformation rules. These proofs are then lifted to a level that is appropriate for end users, by pruning extraneous parts and using templates to generate "surface syntax" that is more palatable than the Sequent Calculus notation.

Borgida's work was followed up by Kwong [8], who implemented it as part of a tool for generating explanations of subsumptions in $\mathcal{ALC}$. An example of the output produced by Kwong's work is shown in Figure 3, which presents an explanation for $\exists hasFriend.\top \sqcap \forall hasFriend.\neg(\exists hasChild.\neg Doctor \sqcup \exists hasChild.Lawyer) \sqsubseteq \exists hasFriend.\forall hasChild.(Rich \sqcup Doctor)$. It is notable that each step contains two subsumptions, with one being derived from the other according to a particular transformation rule.

It is easy to see that approach taken in [3] and in [8] is likely to be more acceptable to end users as a form of explanation than simply extracting and presenting the completion tree from a tableaux reasoner. Indeed reading such a proof probably requires much less training than would be required to understand and read tableaux based proofs. However, the structure and style of proofs such as the one above raise the issue of *proof checking versus proof understanding*.

3.1 Proof Checking versus Understanding

Despite the fact that Natural Deduction based explanations are obviously better than tableaux based proofs, it is not clear how successful these proofs would be for users of tools like Protégé-4. While some justifications suffer from there being too big a gap between the premises and conclusion, it is arguable that Natural Deduction based explanations suffer from the opposite problem. One of the main features of Natural Deduction inspired proof techniques, is that the proofs generated by them can be very fine-grained. This is because each step corresponds to the application of a transformation rule. Each step asserts a simple truth that is easy to verify. While this makes it easy to understand how

—Explanation—

1 $\exists \boldsymbol{hasFriend}.\top \sqcap \forall \boldsymbol{hasFriend}.\neg(\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqcup \exists \boldsymbol{hasChild}.\boldsymbol{Lawyer}) \sqsubseteq \exists \boldsymbol{hasFriend}.\forall \boldsymbol{hasChild}.(\boldsymbol{Rich} \sqcup \boldsymbol{Doctor})$ is true,
because $\top \sqcap \neg(\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqcup \exists \boldsymbol{hasChild}.\boldsymbol{Lawyer}) \sqsubseteq \forall \boldsymbol{hasChild}.(\boldsymbol{Rich} \sqcup \boldsymbol{Doctor})$ is true.
(LEFT-DIAMOND)

2 $\top \sqcap \neg(\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqcup \exists \boldsymbol{hasChild}.\boldsymbol{Lawyer}) \sqsubseteq \forall hasChild.(Rich \sqcup Doctor)$ is true,
because $\top \sqcap \neg\exists \boldsymbol{hasChild}.\boldsymbol{Lawyer} \sqcap \neg\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqsubseteq \forall hasChild.(Rich \sqcup Doctor)$ is true.
(LEFT-NOT-OR)

3 $\top \sqcap \neg\exists \boldsymbol{hasChild}.\boldsymbol{Lawyer} \sqcap \neg\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqsubseteq \forall hasChild.(Rich \sqcup Doctor)$ is true,
because $\neg\exists \boldsymbol{hasChild}.\boldsymbol{Lawyer} \sqcap \neg\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqsubseteq \forall hasChild.(Rich \sqcup Doctor)$ is true.
(SIMPLIFICATION)

4 $\neg\exists \boldsymbol{hasChild}.\boldsymbol{Lawyer} \sqcap \neg\exists \boldsymbol{hasChild}.\neg \boldsymbol{Doctor} \sqsubseteq \forall \boldsymbol{hasChild}.(\boldsymbol{Rich} \sqcup \boldsymbol{Doctor})$ is true,
because $\neg \boldsymbol{Lawyer} \sqcap \neg\neg \boldsymbol{Doctor} \sqsubseteq \boldsymbol{Rich} \sqcup \boldsymbol{Doctor}$ is true.
(RIGHT-BOX)

5 $\neg Lawyer \sqcap \neg\neg \boldsymbol{Doctor} \sqsubseteq Rich \sqcup Doctor$ is true,
because $\neg Lawyer \sqcap \boldsymbol{Doctor} \sqsubseteq Rich \sqcup Doctor$ is true.
(LEFT-NOT-NOT)

6 $\neg Lawyer \sqcap \boldsymbol{Doctor} \sqsubseteq Rich \sqcup \boldsymbol{Doctor}$ is true,
because a conjunction of ***Doctor*** on the left hand side is always more specific than ***Doctor*** whereas a disjunction of ***Doctor*** on the right hand side is always more general than ***Doctor***. Therefore this subsumption must hold.
(EQUIVALENCE)

Fig. 3. An Example Explanation Generated By Kwong's $\mathcal{ALC}$ Subsumption Explanation Generator

to get from one step to the next, and makes it easy to *verify that the conclusion follows from the premises*, it can make it difficult to understand how the steps fit together to form the bigger picture. Indeed, despite the availability of a proof, the relationship between the premises and the conclusion can still be non-obvious.

One way to address this problem is to add more transformation rules, which group together or bridge other rules. In this case, it is necessary to devise a set of derived rules that make Natural Deduction style proofs nicer to read. On the flip side, a related approach, is to omit *trivial* inference steps that can otherwise be inferred by the person reading the proof [9, 6]. In this case it is necessary to identify what constitutes a trivial inference step. Both approaches point to some middle ground, where each step comfortably takes a reader from a set of premises or intermediate formulae to an intermediate conclusion.

Therefore, our approach is to construct proofs from justifications, where each step in the proof is not defined by a transformation rule, but is defined by a justification—i.e. entailment. In this scenario, each step represents a *manageable* chunk of information that can be easily traversed by a user, but is not so fine-grained that it clouds the overall structure of the proof and how the premises (asserted axioms) relate to the conclusion (entailment).

3.2 Proofs Based On Justifications

In addition to the issues discussed above, there are several other well founded reasons for basing proofs on justifications:

One, when a user seeks an explanation for an entailment in an ontology, they do so *not to verify that the entailment holds*, but to determine *what it is they have said in the ontology* that causes the entailment to hold, and *why these statements cause the entailment to hold*. Justifications have been used very effectively as a way of addressing the "what causes it?" problem. Indeed, by definition, they address this problem directly. To a large degree, justifications can also successfully be used to address the "why do these axioms cause it?" problem—the study detailed in [5] showed that there are many kinds of justifications that people *can understand*. This includes people who have never seen justifications before, or people who have minimal experience with OWL and Description Logics. People quickly feel comfortable in understanding how justifications work, and, after seeing just two or three example justifications, they generally feel comfortable with reading them. Note that this is despite the fact that justifications are axiom oriented and many modern ontology editors such as Protégé-4, the NeOn toolkit and Top Braid Composer present ontologies in a frame-based fashion. Hence, in cases where people can understand justifications, it would seem strange and unnecessary to turn these justifications into proofs similar to the ones presented above.

Two, proofs based on justifications are *reasoning procedure independent*. This is because *each "step" in the proof is based on the entailment relation* and not some transformation rule that is sensitive to the syntax of the axioms in the step. There is no explicit fixed set of rules for deriving one axiom from a set of axioms. This means that the technique will work for different logics and future OWL extensions.

Three, justifications are *reasoner independent*. This means that a service that computes justification oriented proofs can be designed that can be used with arbitrary reasoners. People can use their favourite reasoner and they are not constrained to using a particular reasoner, or type of reasoner, just to obtain the proofs.

4 Justification Oriented Proofs

The main idea behind a justification oriented proof is depicted in Figure 4. The numbered rectangles represent axioms, with the rightmost rectangle, labelled η, representing the entailment of interest. The shaded rectangles labelled with "1" – "6" represent exactly the axioms that appear in the original justification $\mathcal{J}$ for the entailment (and are therefore in the ontology as asserted axioms). Hence $\mathcal{J} = \{1, 2, 3, 4, 5, 6\}$ is a justification for η with respect to the ontology that entails η. The idea behind a justification oriented proof, is to augment a justification with helpful intermediate steps, called *lemmas*. This produces a weakly connected directed acyclic graph, with one sink node that represents the entailment of interest and a source node for each axiom in the justification. In the example shown in Figure 4, axiom 7 is a lemma for axioms 1, 2 and 3 (conversely, axioms 1, 2 and 3 are a justification for axiom 7). Axiom 8 is a lemma for axioms 3, 4 and 5 (conversely axioms 3, 4 and 5 are a justification for axiom 8). Together

axioms 6, 7 and 8 constitute a justification for η i.e. the entailment. Notice that axiom 3 participates in different justifications for different lemmas. The main idea of this approach, is that compared to the justification $\mathcal{J}$, it is easier to see and understand why axioms 6, 7 and 8 entail η, and it is also easy to see why axioms 1, 2 and 3 entail axiom 7, and why axioms 3, 4 and 5 entail axiom 8. Moreover, the ability to spot that $\mathcal{J}$ entails axioms 7 and 8, and to realise the part these axioms play, may be key to understanding how $\mathcal{J} \models \eta$, yet they may not be at all salient to a person looking at $\mathcal{J}$. In other words, the process of understanding why a justification supports an entailment, is transformed to understanding how subsets of the justification result in intermediate entailments, and understanding how these intermediate entailments fit together to give rise the main entailment of interest.

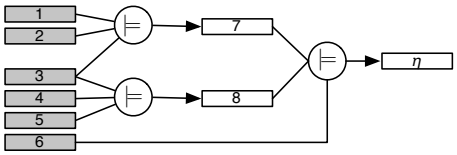

Fig. 4. A schematic of a Justification Oriented Proof

4.1 Choosing The Steps

Key to producing a "good" justification oriented proof is good choice of the intermediate lemmas and their lemmatised justifications. In the framework presented here, a *complexity model* that predicts how difficult or easy a justification is to understand is used. The complexity model used is the one that is described in previous work [5]. It is used to construct proofs where each lemmatised justification is deemed easy to understand. Proofs are computed in a top down manner, where, for a justification $\mathcal{J}$ for η, a lemmatised justification $\mathcal{J}'$ of low complexity, as predicted by the model, is computed for η. Then, for each axiom λ in $\mathcal{J}'$ that is not in $\mathcal{J}$, a justification $\mathcal{J}''$ for λ with respect to $\mathcal{J}$ is computed. If $\mathcal{J}''$ is of too high complexity then it is lemmatised and the process is repeated. An example justification oriented proof for the justification shown in Figure 1 is shown in Figure 5, and an example justification oriented proof for the justification shown in Figure 2 is shown in Figure 6.

There are some further restrictions on how lemmas in justification oriented proofs can be generated from the original justification. These restrictions are discussed in more detail in [5], however, the most pertinent restriction is that lemmas must either be of the form $\top \sqsubseteq A$ or $A \sqsubseteq \bot$ for some A in the signature of the justification, or must be drawn from the deductive closure of *tidy* subsets of the original justification. A set of axioms is *tidy* is it is (a) consistent (b) entails no synonyms of $\top$ ($\top \sqsubseteq A$ for any A in the signature of the original justification), and (c) entails no synonyms of $\bot$ ($A \sqsubseteq \bot$ for any A in the signature of the original justification). These restrictions are in place to avoid counter-intuitive lemmatisations, examples of which may be found in [5].

Entailment : $\mathsf{Person} \sqsubseteq \bot$

$\mathbf{Person} \sqsubseteq \neg\mathbf{Movie}$ (1)

$\top \sqsubseteq \mathsf{Movie}$ (2)

$\quad \forall \mathsf{hasViolenceLevel}.\bot \sqsubseteq \mathsf{Movie}$ (3)

$\quad\quad \forall \mathsf{hasViolenceLevel}.\bot \sqsubseteq \mathsf{RRated}$ (4)

$\quad\quad\quad \mathbf{RRated} \equiv (\exists \mathbf{hasScript}.\mathbf{ThrillerScript}) \sqcup (\forall \mathbf{hasViolenceLevel}.\mathbf{High})$ (5)

$\quad\quad \mathsf{RRated} \sqsubseteq \mathsf{Movie}$ (6)

$\quad\quad\quad \mathbf{RRated} \sqsubseteq \mathbf{CatMovie}$ (7)

$\quad\quad\quad \mathbf{CatMovie} \sqsubseteq \mathbf{Movie}$ (8)

$\quad \exists \mathsf{hasViolenceLevel}.\top \sqsubseteq \mathsf{Movie}$ (9)

$\quad\quad \mathbf{Domain}(\mathbf{hasViolenceLevel}, \mathbf{Movie})$ (10)

Fig. 5. A schematic of a justification oriented proof for the justification shown in Figure 1

4.2 Examples

In what follows, two examples relating to the justifications shown in Figures 1 and 2 are presented. Note that the presentation style used here is merely for illustrative purposes, it is designed to give a flavour of the kinds of lemmas that get introduced into a proof rather than as an end user presentation device. The axioms shown in bold are the axioms that appear in the original justification, and are therefore asserted, and all other axioms correspond to lemmas. Notice that axioms at each level of indentation form a justification for the axiom that is above them.

In the presentation here, the example shown in Figure 5 may be read as follows. Person is unsatisfiable because Person is disjoint with Movie (axiom 1, asserted, and thus bold) and yet everything must be a Movie (axiom 2, a lemma, generated by our approach). The level below $\top \sqsubseteq \mathsf{Movie}$, corresponding to axiom 3 and axiom 9, explains why everything must be a Movie. This is due to the fact that everything that does not have a violence level is a Movie (axiom 3), and everything that does have a violence level is a Movie (axiom 9). The reason that anything that does not have a violence level is a Movie is due to axioms 4 and 6, both of which are lemmas, which form a justification saying that anything that does not have a violence level is RRated and anything that is RRated is a Movie. Axiom 4, which specifies that anything that does not have a violence level is RRated, is a lemma which is entailed by one asserted axiom, i.e. axiom 5. Similarly, the lemma corresponding to axiom 6 is entailed by two asserted axioms, 7 and 8. Finally, the lemma that everything that has a violence level is a Movie is entailed by axiom 10, the asserted domain axiom.

Figure 6 shows a justification oriented proof for the entailment DailyMirror is a Newspaper, with a regular justification for this entailment being shown in Figure 2. Notice that the summarising effect of the justification oriented proof is rather dramatic—in the top level justification in the proof there are three

Entailment : Newspaper(DailyMirror)

---

reads(**Mick DailyMirror**)
Tabloid $\sqsubseteq$ **Newspaper**
$\forall$reads.Tabloid(Mick)
  WhiteVanMan $\sqsubseteq$ $\forall$**reads.Tabloid**
  WhiteVanMan(Mick)
    Man $\sqcap$ $\exists$drives.(Van $\sqcap$ WhiteThing) $\sqsubseteq$ WhiteVanMan
      WhiteVanMan $\equiv$ **Man** $\sqcap$ $\exists$**drives**.(**Van** $\sqcap$ **WhiteThing**)
    WhiteThing(**Q123ABC**)
    drives(**Mick Q123ABC**)
    Man(Mick)
      Adult $\sqcap$ Male $\sqcap$ Person $\sqsubseteq$ Man
        Man $\equiv$ **Adult** $\sqcap$ **Male** $\sqcap$ **Person**
      Adult(Mick)
        ...
      Male(**Mick**)
      Person(Mick)
        hasPet(Mick Rex)
          ...
        Domain(**hasPet Person**)

Fig. 6. A schematic of a justification oriented proof for the justification shown in Figure 2. Note that the full proof is not shown here (see dots) due to space limitations

summarising axioms (compared with 15 axioms in the original justification), two of which have been asserted and were therefore in the original justification, reads(Mick DailyMirror) and Tabloid $\sqsubseteq$ Newspaper, and one lemma, namely, $\forall$reads.Tabloid(Mick) i.e. Mick only reads Tabloids. This lemma is entailed by a justification containing two axioms, one of which is asserted, WhiteVanMan $\sqsubseteq$ $\forall$reads.Tabloid, and one of which is another lemma, WhiteVanMan(Mick).

While the presentation in Figures 5 and 6 is for illustrative purposes only, one could imagine an interactive debugging and explanation tool that uses the underlying proof structures to derive a user friendly proof presentation. Such a presentation may offer the ability to expand and collapse various levels. Additionally, the presentation used here is a "top down" presentation which goes from entailment to asserted axioms. The alternative presentation would be a "bottom up" presentation that would go from asserted axioms, via lemmas, to the entailment.

Given the proofs above, it is noticeable that general concept inclusion axioms (GCIs) are introduced into some of the intermediate justifications. For example, the GCI, Man $\sqcap$ $\exists$drives.(Van $\sqcap$ WhiteThing) $\sqsubseteq$ WhiteVanMan is introduced as a lemma into the proof in Figure 6. Since people typically edit ontologies in environments that show entities using frame-based displays, such as Protégé-4, a natural question that arises is, whether people who use these environments would

find it acceptable to see GCIs in proofs. In the study carried out in [5], it was found that people who were unused to seeing GCIs, were initially surprised at seeing one for the first time, with comments such as, "I've never seen a subclass statement this way round before". However, in terms of reading an understanding justifications, once over the initial surprise, none of the participants found GCIs problematic.

5 Conclusions and Future Work

Some justifications can be difficult for people to understand. Justification oriented proofs have been presented as a possible solution to this problem. A justification oriented proof is composed of a series of justifications, which are lemmatisations of the original justification. In the framework presented in this paper, justification oriented proofs are derived using a complexity model, which predicts how easy or difficult it is to understand a justification, and by selection of lemmas that follow from subsets of the original justification. As future work, we aim to investigate how sensitive the proofs are to the complexity model that drives their construction. We also intend to carry out studies that show if justification oriented proofs are beneficial to users of tools such as Protégé-4.

References

1. Franz Baader and Bernhard Hollunder. Embedding defaults into terminological knowledge representation formalisms. *J. Autom. Reasoning*, 14(1):149–180, 1995.
2. Franz Baader and Boontawee Suntisrivaraporn. Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In *Proceedings of the 3rd Knowledge Representation in Medicine Conference (KR-MED'08): Representing and Sharing Knowledge Using SNOMED*, 2008.
3. Alex Borgida, Enrico Franconi, and Ian Horrocks. Explaining $\mathcal{ALC}$ subsumption. In *Proc. of the 14th Eur. Conf. on Artificial Intelligence (ECAI 2000)*, pages 209–213. IOS Press, 2000.
4. Gerhard Gentzen. Untersuchungen über das logische schließen II. *Mathematische Zeitschrift*, 39, 1935.
5. Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Lemmas for justifications in owl. In *Description Logics (DL 2009)*, 2009.
6. Xiaorong Huang. Reconstructing proofs at the assertion level. In *12th Conference on Automated Deduction (CADE)*, 1994.
7. Aditya Kalyanpur. *Debugging and Repair of OWL Ontologies*. PhD thesis, The Graduate School of the University of Maryland, 2006.
8. Francis King Hei Kwong. Practical approach to explaining $\mathcal{ALC}$ subsumption. Technical report, The University of Manchester, 2005.
9. Christoph Lingenfelder. Structuring computer generated proofs. In *International Joint Conference on Artificial Intelligence*, 1989.
10. Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *IJCAI International Joint Conference on Artificial Intelligence*, 2003.