Analysing Multiple Versions of an Ontology: A Study of the NCI Thesaurus

Rafael S. Gonçalves, Bijan Parsia, and Uli Sattler

School of Computer Science, University of Manchester, UK
{goncalvj,bparsia,sattler}@cs.man.ac.uk

Abstract. The detection of changes between OWL ontologies is an important service for ontology engineering. There are several approaches to this problem, both syntactic and semantic. A purely syntactic analysis of changes is insufficient to detect changes with logical effect, while the current state of the art in semantic diffing ignores logically ineffectual changes, which might be of great interest to the user. We develop an exhaustive categorisation of ineffectual changes, based on their justifications. In order to verify the applicability of our approach, we collected 88 OWL versions of the National Cancer Institute (NCI) Thesaurus (NCIt), and extracted all pairwise, consecutive diffs. We discovered a substantial number of ineffectual changes and, as a result, argue that the devised categorisation of changes is beneficial for ontology engineers. We devised and applied a method for performance impact analysis (*culprit finding*) based on the diff between ontologies, and identified a number of culprits between two NCIt versions.

1 Motivation

The comparison of ontologies is a valuable service whether for purely analytic purposes, versioning systems [3], or collaboration. When comparing two ontologies it is desirable to detect both syntactic and logical changes. OWL defines a high level notion of syntactic equivalence, so-called "structural equivalence", which abstracts from such concrete details as the order of axioms. Associated with structural equivalence is *structural difference*. A different syntactic approach is that of an edit-based diff, wherein change records are produced within the ontology editor being used thereby capturing the history of change, as implemented in Swoop [8]. The diffs mentioned so far, as well as PROMPTDIFF [12], do not recognize the logical impact of changes. When analysing the impact of changes, it is sensible to inspect not only logically effectual changes, but also ineffectual ones since these might have been intended to have logical impact, and thus may be of interest to users. Semantic diffs, such as CEX [10, 4], OWLDiff [11] or ContentCVS [7] detect only effectual changes. So on the one hand, syntactic diffs detect without distinction both effectual and ineffectual changes, and on the other hand semantic diffs do not analyse ineffectual changes.

In this paper we propose a diff notion that builds on structural diff with a logical impact analysis, which we refer to as *intentional difference*, incorporating a categorisation of ineffectual axioms based on their justifications. The goal of this categorisation is to suggest on the intent behind such changes. For the purpose of verifying the suitability of our approach, we collected all 88 versions of the National Cancer Institute (NCI)

Thesaurus (NCIt) available in OWL format, freely downloadable[1] from the web, and conducted a diachronic study of the corpus. This study consisted of the extraction of all pairwise, consecutive diffs between NCIt versions. Our diff revealed a fairly high number of ineffectual changes across the corpus, averaging at 13% and even reaching values above 90%. In addition to this we carried out a reasoner performance test to inspect the performance impact of both effectual and ineffectual changes throughout the NCIt. While ineffectual changes carry no logical impact, it is still the case that they have a performance impact.[2] The test revealed an unusual performance increase between 2 versions, the latter of which was 89% faster and also slightly bigger in number of axioms. This motivated a more in-depth performance impact analysis, wherein we attempt to find subsets of the slow ontology without which the ontology performs considerably faster (referred to as *culprits*). We devise a culprit finding method based on the diff between ontologies, and demonstrate its applicability with a number of culprits for the NCIt case.

2 Preliminaries

We assume the reader to be reasonably familiar with OWL [13], as well as the underlying description logics (DLs) [5], though detailed knowledge is not required. We do use the notion of entailment [2], which is identical to the standard first order logic entailment (albeit restricted to certain syntactic forms for consequences, typically atomic subsumption). When comparing two versions of an ontology we refer to the earlier version as $\mathcal{O}_1$, and the more recent as $\mathcal{O}_2$. A justification $\mathcal{J}$ of a consequence α is a minimal subset of an ontology $\mathcal{O}$ that is sufficient for α to hold [9]. The signature of an ontology $\mathcal{O}$ is denoted $\widetilde{\mathcal{O}}$. An axiom $\alpha \in \mathcal{O}_1$ is logically ineffectual for an ontology $\mathcal{O}_2$ iff $\alpha \notin \mathcal{O}_2$ and $\mathcal{O}_2 \models \alpha$, and we often describe it as having no impact.

3 Ontology Difference

The problem of computing the difference between pairs of ontologies has been approached both syntactically and semantically. We distinguish two major aspects of ontology diffing: *(i)* the detection of changes, and *(ii)* the presentation of changes to the end-user. As we analyse existing diff approaches, we point out that most effort has been largely dedicated to *(i)*. It is often the case that the output of diff operations is the set of axioms or terms in the diff. While this may reflect the desired identification of change, it does not convey sufficient information to the user w.r.t. the intent of changes, or whether these are effectual or not.

3.1 Diff Desiderata

Table 1 summarises useful features of an ontology diff, and whether existing approaches exhibit such desiderata.

[1] `http://evs.nci.nih.gov/ftp1/NCI_Thesaurus`

[2] A trivial example is adding all inferred subsumptions, therefore speeding up reasoning tasks.

Table 1. Desiderata of ontology diffing approaches.

Properties	**ContentCVS**	**CEX**	**OWLDiff [11]**	**PROMPTDIFF**	**Swoop Diff**
Difference Detection					
Syntactic analysis	✓	✗	✓	✓	✓
Semantic analysis	✓	✓	✓	✗	✗
Effectively computable	✓	✗	✓	✓	✓
OWL 2 adequacy	✓	✗	✓	✗	✓
Difference Output					
Axiom-based	✓	✓	✓	✗	✓
Term-based	✗	✓	✗	✓	✓
Effectual change analysis	✓	✓	✓	N/A	N/A
Ineffectual change analysis	✗	✗	✗	N/A	N/A

Among the stated properties, an ideal logical diff should combine effective computability for OWL 2 ontologies while providing some analysis of the impact of changes, whether these be effectual or ineffectual. Although this is a complex task in itself, from Table 1 we see that some diffs analyse effectual changes, but none of them inspects ineffectual changes. This desideratum leads to the categorisation method proposed in this paper for the latter type of changes.

3.2 Intentional Diff

Given the limitations of diff approaches described in Table 1 w.r.t. (ii) (as described at the beginning of Section 3), we build on the notion of structural difference with a categorisation mechanism for ineffectual axioms. This requires checking if axioms in the first ontology are entailed by the second (and vice-versa), if that is not the case then those axioms are regarded as effectual changes.

Consider the following ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, which are referred to in examples throughout this section:

$$\begin{aligned}\mathcal{O}_1 = \{\alpha_1 :&\quad A \sqsubseteq C,\\ \alpha_2 :&\quad B \sqsubseteq C,\\ \alpha_3 :&\quad E \equiv D,\\ \alpha_4 :&\quad D \sqsubseteq F,\\ \alpha_5 :&\quad F \sqsubseteq G,\\ \alpha_6 :&\quad G \sqsubseteq H \sqcap \exists s.H,\\ \alpha_7 :&\quad F \sqsubseteq I,\\ \alpha_8 :&\quad F \sqsubseteq G \sqcap I \sqcap J\}\end{aligned}$$

$$\begin{aligned}\mathcal{O}_2 = \{\beta_1 :&\quad A \sqsubseteq B \sqcup C,\\ \beta_2 :&\quad A \sqsubseteq B,\\ \beta_3 :&\quad B \sqsubseteq C,\\ \beta_4 :&\quad E \sqsubseteq D,\\ \beta_5 :&\quad D \sqsubseteq E,\\ \beta_6 :&\quad E \sqsubseteq B \sqcup \exists r.C,\\ \beta_7 :&\quad D \sqsubseteq E \sqcup G,\\ \beta_8 :&\quad G \sqsubseteq \exists s.H \sqcap H,\\ \beta_9 :&\quad F \sqsubseteq G \sqcap I\}\end{aligned}$$

The notion of structural difference is based on OWL's notion of structural equivalence (denoted $\equiv_s$) [13]. The latter deems the order of axioms in an ontology as irrele-

[2] For DLs up to $\mathcal{SROIQ}$.

vant, as well as the order of disjunctions or conjunctions between concepts. Therefore one can rule out differences that an otherwise syntactic equality based diff would detect.

Definition 1 (Structural Difference [6]) *The structural difference between $\mathcal{O}_1$ and $\mathcal{O}_2$ are the following sets:*

- $\mathrm{Additions}(\mathcal{O}_1, \mathcal{O}_2) = \{\beta \in \mathcal{O}_2 \mid$ *there is no* $\alpha \in \mathcal{O}_1$ *s.t.* $\alpha \equiv_s \beta\}$
- $\mathrm{Removals}(\mathcal{O}_1, \mathcal{O}_2) = \{\alpha \in \mathcal{O}_1 \mid$ *there is no* $\beta \in \mathcal{O}_2$ *s.t.* $\alpha \equiv_s \beta\}$

So if there is an axiom β s.t. $\beta \in \mathrm{Additions}$, this implies that $\beta \in \mathcal{O}_2 \setminus \mathcal{O}_1$, and similarly for $\mathrm{Removals}$. Examine the following example:

Example 1 *From the defined ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ we have that:*

- $\diamond$ $\mathrm{Additions}(\mathcal{O}_1, \mathcal{O}_2) = \{\beta_1, \beta_2, \beta_4, \beta_5, \beta_6, \beta_7, \beta_9\}$
- $\diamond$ $\mathrm{Removals}(\mathcal{O}_1, \mathcal{O}_2) = \{\alpha_1, \alpha_3, \alpha_4, \alpha_5, \alpha_7, \alpha_8\}$

Note that the axiom α_2 is syntactically equal to β_3; $\alpha_2 = \beta_3$. We also have that $\alpha_6 \equiv_s \beta_8$. Therefore these axioms are not reported as changes.

Based on these two sets, the logical difference pinpoints which axioms in $\mathrm{Additions}$ (or $\mathrm{Removals}$) affect the set of entailments of $\mathcal{O}_1$ (or $\mathcal{O}_2$). In other words, it distinguishes between those axioms in the structural difference which are entailed by $\mathcal{O}_1$ (or $\mathcal{O}_2$), as follows:

Definition 2 (Logical Difference) *The logical difference between $\mathcal{O}_1$ and $\mathcal{O}_2$ are the following sets:*

- $\mathrm{EffectualAdditions}(\mathcal{O}_1, \mathcal{O}_2) = \{\beta \in \mathrm{Additions}(\mathcal{O}_1, \mathcal{O}_2) \mid \mathcal{O}_1 \not\models \beta\}$
- $\mathrm{EffectualRemovals}(\mathcal{O}_1, \mathcal{O}_2) = \{\alpha \in \mathrm{Removals}(\mathcal{O}_1, \mathcal{O}_2) \mid \mathcal{O}_2 \not\models \alpha\}$
- $\mathrm{IneffectualAdditions}(\mathcal{O}_1, \mathcal{O}_2) = \mathrm{Additions} \setminus \mathrm{EffectualAdditions}$
- $\mathrm{IneffectualRemovals}(\mathcal{O}_1, \mathcal{O}_2) = \mathrm{Removals} \setminus \mathrm{EffectualRemovals}$

The resulting sets $\mathrm{IneffectualAdditions}$ and $\mathrm{IneffectualRemovals}$ are composed of those axioms which do not change the set of entailments of $\mathcal{O}_1$ and $\mathcal{O}_2$, respectively. An axiom β is in $\mathrm{IneffectualAdditions}$ iff $\mathcal{O}_1 \models \beta$, and similarly for $\mathrm{IneffectualRemovals}$ (Example 2).

Example 2 *Given the sets* $\mathrm{Additions}$ *and* $\mathrm{Removals}$ *(from Example 1) we have that:*

- $\diamond$ $\mathrm{EffectualAdditions}(\mathcal{O}_1, \mathcal{O}_2) = \{\beta_2, \beta_6\}$
- $\diamond$ $\mathrm{EffectualRemovals}(\mathcal{O}_1, \mathcal{O}_2) = \{\alpha_4, \alpha_8\}$
- $\diamond$ $\mathrm{IneffectualAdditions}(\mathcal{O}_1, \mathcal{O}_2) = \{\beta_1, \beta_4, \beta_5, \beta_7, \beta_9\}$
- $\diamond$ $\mathrm{IneffectualRemovals}(\mathcal{O}_1, \mathcal{O}_2) = \{\alpha_1, \alpha_3, \alpha_5, \alpha_7\}$

In order to characterise ineffectual changes, we devise a categorisation of axioms based on their justifications as follows:

Definition 3 (Intentional difference) *An axiom $\alpha \in \mathrm{IneffectualRemovals}$ is:*

- Strengthened, *if there is a* $\mathcal{J}$ *for* α *with* $\mathcal{J} \cap \text{EffectualAdditions} \neq \emptyset$.
- Rewritten, *if there is a justification* $\mathcal{J}$ *for* α *with* $\mathcal{J} \cap \text{Additions} \neq \emptyset$, *and* $\alpha \models \mathcal{J}$. *If* $\mathcal{J} \subseteq \text{Additions}$ *then* α *is a complete rewrite, otherwise a partial rewrite.*
- Redundant, *if there is a* $\mathcal{J}$ *for* α *with* $\mathcal{J} \subseteq (\mathcal{O}_1 \cap \mathcal{O}_2)$. *If* $\mathcal{J} \subseteq (\mathcal{O}_1 \cap \mathcal{O}_2) \cup \text{IneffectualAdditions}$ *then* α *is an avoided redundancy.*

To obtain the corresponding categories for added axioms $\beta \in \text{IneffectualAdditions}$, *replace* α, Additions, EffectualAdditions *and* IneffectualAdditions *with* β, Removals, EffectualRemovals *and* IneffectualRemovals *respectively. In* IneffectualAdditions *the label for the criteria of* Strengthened *axioms changes to* Weakened *axioms.*

The intentional difference gives possibly overlapping sets of axioms, as demonstrated in Example 3. Also we note that these categories are exhaustive, in the sense that there is no axiom such that the justifications of which do not imply one of the defined categories. Consider an axiom α and ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, with $\alpha \in \mathcal{O}_1$ but $\alpha \notin \mathcal{O}_2$, and $\mathcal{O}_2 \models \alpha$. Then there must be a justification $\mathcal{J} \subseteq \mathcal{O}_2$ for α. If $\mathcal{J} \subseteq (\mathcal{O}_1 \cap \mathcal{O}_2) \cup \text{IneffectualAdditions}$ then α is redundant, otherwise if $\mathcal{J} \cap \text{EffectualAdditions} \neq \emptyset$, then α is strengthened.

Example 3 *Given the sets* IneffectualAdditions *and* IneffectualRemovals *(from Example 2) we have that:*

$\mathcal{O}_1 \longrightarrow \mathcal{O}_2$	$\mathcal{O}_2 \longrightarrow \mathcal{O}_1$
$\diamond$ Rewritten $= \{\alpha_3\}$	$\diamond$ Rewritten $= \{\beta_9\}$
$\diamond$ Strengthened $= \{\alpha_1\}$	$\diamond$ Weakened $= \{\beta_7, \beta_9\}$
$\diamond$ Redundant $= \{\alpha_1, \alpha_3, \alpha_5, \alpha_7\}$	$\diamond$ Redundant $= \{\beta_1, \beta_4, \beta_5, \beta_7, \beta_9\}$

Note that the existence of a rewritten axiom from $\mathcal{O}_1$ *to* $\mathcal{O}_2$ *does not imply that the same holds in the opposite direction. This is applicable to all categories. Also we can have that an axiom is in more than one categorical set, exemplified as follows:*

Rewritten and redundant *The axiom* α_3 *has been rewritten from* $\mathcal{O}_1$ *to* $\mathcal{O}_2$. *The justification for* α_3 *is* $\mathcal{J}_1 = \{\beta_4, \beta_5\}$, *which is categorised as a rewrite since* $\alpha_3 \models \mathcal{J}_1$. *However, since* $\{\beta_4, \beta_5\} \in \text{IneffectualAdditions}$, $\mathcal{J}_1$ *also indicates a redundancy. So the axiom* α_3 *is part rewritten part redundant.*

Strengthened and redundant *Consider axiom* α_1; *we can see that* $\mathcal{O}_2 \models \alpha_1$. *A justification* $\mathcal{J}_1$ *for* α_1 *is* $\mathcal{J}_1 = \{\beta_2, \beta_3\}$, *which indicates a strengthening (since* $\beta_2 \in \text{EffectualAdditions}$*), as well as a redundancy (*$\beta_3 \in \mathcal{O}_1 \cap \mathcal{O}_2$*). Another justification* $\mathcal{J}_2 = \{\beta_1, \beta_3\}$ *indicates a strict redundancy;* $\beta_1 \in \text{IneffectualAdditions}$.

Rewritten, weakened and redundant *Axiom* β_9 *is categorised as rewritten, weakened and redundant. A justification for* β_9 *is* $\mathcal{J}_1 = \{\alpha_5, \alpha_7\}$, *where* $\beta_9 \models \mathcal{J}_1$, *pointing to a rewrite. We also have that* $\{\alpha_5, \alpha_7\} \in \text{IneffectualRemovals}$, *therefore being categorised as redundant as well. A second justification is* $\mathcal{J}_2 = \{\alpha_8\}$, *and since* $\alpha_8 \in \text{EffectualRemovals}$, β_9 *is categorised as weakened.*

While the logical diff identifies those logically ineffectual axioms in the difference, it does not suggest on the intent of change or present appropriate reasons for it, i.e. justifications. With the categorisation method described, users have, at the very least, an

indicator as to why such axioms have no impact. Note that these categories are merely suggestive of the developers' intent. In order to ensure the real intent one would require either a detailed edit-based diff or contact with the ontology developers.

4 Empirical results

In order to substantiate our approach to ontology diffing, we carried out a diachronic study of the NCIt using the methods described. The NCIt archive[3] contains 88 versions of the ontology in OWL format, two of which were unparsable (releases 05.03F and 05.04d) with the OWL API,[4] and consequently Protégé.[5] The experiment machine is an Intel Xeon Quad-Core 3.20GHz, with 12Gb DDR3 RAM dedicated to the Java Virtual Machine (JVM v1.5). The system runs Mac OS X 10.6.7, and all tests were run using the OWL API (v3.1). All gathered test data is available from `http://owl.cs.manchester.ac.uk/ncit`, a part of it is published on Google Public Data Explorer,[6] and can be visualised at `http://bit.ly/jFKU3R`.

4.1 Axioms Difference

The logical difference throughout the NCIt time-line consists mostly of subclass axioms (see Figure 1, and for complete results the mentioned website), with an average of 75% (excluding $\mathcal{O}_{14}$ and $\mathcal{O}_{16}$). The average proportion of logical changes is 15%, and the remaining are annotation changes. It should be noted that, despite the large number of annotations, NCIt developers devoted considerable effort towards the logical part of the ontology. Version $\mathcal{O}_6$ is a curious case, where a large number of classes (5170) were renamed,[7] and around 220,000 annotations and 14,418 subclass axioms were deleted. This indicates a possible re-modelling, or mass-renaming of classes in the NCIt at this point. More evidence to support this includes the addition of 30,859 subclass axioms, 9,070 classes and 23 object properties (and roughly 240,000 entity annotations). Similarly in $\mathcal{O}_{25}$ a series of changes were carried out to the subsumption hierarchy, with the removal of 8,231 subclass axioms and 2,899 equivalent class axioms compared to the previous version, and also the addition of 10,591 subclass axioms and 3,011 equivalent class axioms.

There is a fair amount of ineffectual removals in the corpus, reaching values of 93% in $\mathcal{O}_{29}$ or 97% in $\mathcal{O}_{16}$, and with an average of 35% of all logical removals (see Figure 1). Out of these ineffectual removals 92% turned out to be strengthened axioms (e.g. $\mathcal{O}_{27}$ has 3,104 strengthened axioms out of 3,843 removals), while 42% were removed redundancies. On average 5% of logical additions are ineffectual, yet there are some high values such as 61% in $\mathcal{O}_{24}$. Among these 73% are added redundancies, and 82% are weakened axioms. We also identified a number of rewrites in the corpus. Particularly

[3] `http://evs.nci.nih.gov/ftp1/NCI_Thesaurus`
[4] `http://owlapi.sourceforge.net/`
[5] `http://protege.stanford.edu/`
[6] `http://www.google.com/publicdata/home`
[7] Since throughout the NCIt evolution no classes are removed.

from $\mathcal{O}_{32}$ to $\mathcal{O}_{33}$ there are 227 rewritten axioms, typically taking a form as shown in Example 4.

Example 4 $A \equiv B \sqcap (\exists r.D) \sqcap (\exists s.F) \sqcap (\forall t.G)$ *rewritten into:*
$A \equiv B \sqcap ((\exists r.D) \sqcap (\exists s.F)) \sqcap (\forall t.G)$

This kind of change is not only syntactic but also trivial and easily detected. While ideally the underlying structural diff would not include these, at least with our categorisation and alignment with source axioms, it is easy to spot and recognize the triviality. One can also argue that certain ineffectual changes are in fact refactorings of one version into another, albeit in the case of strengthened and weakened axioms one could say that the intention was exactly that but turned out not to have the desired effect. The distinction here should be made that the strengthening of an axiom does not necessarily mean strengthening of the ontology. Consider an ontology $\mathcal{O}_1 = \{\alpha_1 : A \sqsubseteq B, \alpha_2 : A \sqsubseteq C\}$, and a change of α_1 into $A \sqsubseteq B \sqcap C$. The axiom α_1 was strengthened, but the resulting ontology $\mathcal{O}_2 = \{\alpha_1 : A \sqsubseteq B \sqcap C, \alpha_2 : A \sqsubseteq C\}$ was not. However, if we change $\alpha_2 \in \mathcal{O}_2$ into $A \sqsubseteq C \sqcap D$, then we can say both the axiom α_2 and the ontology $\mathcal{O}_2$ are strengthened.

We noted a recurring trend throughout the NCIt corpus, which is the addition of redundancies. This trend has more incidence up until $\mathcal{O}_8$, but there are high values in the rest of the corpus as well, such as $\mathcal{O}_{35}$ with 174 added redundant axioms (see Figure 1). The highest value found is in $\mathcal{O}_{17}$, where 482 redundant axioms were added. Upon investigating this phenomenon, we found that such added redundancies are, in most or all cases, entailments from previous versions. These entailments are those derived from the transitivity of the subclass relationship, e.g. $\mathcal{O}_1 = \{\alpha_1 : A \sqsubseteq \exists r.B, \alpha_2 : C \sqsubseteq A\}$, $\mathcal{O}_2 = \{\alpha_1, \alpha_2, \alpha_3 : C \sqsubseteq \exists r.B\}$. From the example we see that α_3 is redundant; $C \sqsubseteq A$ suffices for $C \sqsubseteq \exists r.B$ to hold.

Overall the average of ineffectual changes is 13%, while the remaining are effectual. However there are cases where the number of ineffectual changes is quite high, such as $\mathcal{O}_{24}$ where 52% of logical changes are ineffectual, as well as $\mathcal{O}_{27}$, $\mathcal{O}_{29}$ and $\mathcal{O}_{30}$ with 48% each. In retrospect this is a high amount of changes that would go unexplained by existing diffs, and while structural diff captures this it does not analyse the logical impact of such changes.

4.2 Reasoner Performance

It is often the case that, for reasoner testing, only a few or even one ontology version is tested against. There is no reported reasoner benchmark using a corpus of the same kind as the one here described. So, in the process of analysing the NCIt, we evaluated how modern reasoners handle all published OWL versions of the NCIt. Three major DL reasoners were put to the test; FaCT++ (v1.5.1), Pellet (v2.2.2) and HermiT (v1.3.3). Since we also possess the axioms in the difference between NCIt versions, this allows us to test incremental reasoning as well.[8] In Figure 2 we plot the reasoning times in a

[8] As implemented within Pellet.

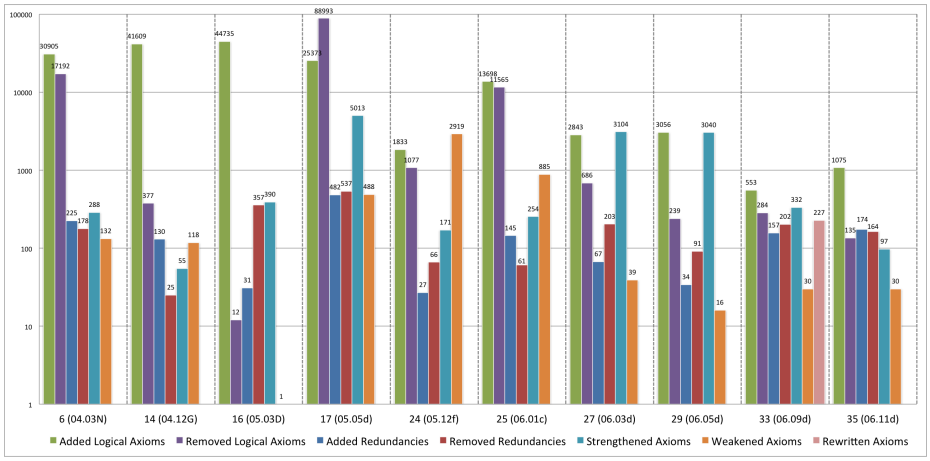

Fig. 1. Logical diff across selected versions of the NCIt (number of axioms).

logarithmic scale of each reasoner, comprising consistency checking, classification and concept satisfiability (denoted $\mathrm{RT}(\mathcal{O})$). Out of the three reasoners put to test, FaCT++ behaves consistently faster than Pellet and HermiT ($\mathcal{O}_{14}$ and $\mathcal{O}_{16}$ aside).

This performance test also shows that, to some degree, incremental reasoning provides a big advantage when handling the NCIt (or other large ontologies) in terms of reasoning time. However it did not terminate upon classifying $\mathcal{O}_{14}$ and, like HermiT,[9] $\mathcal{O}_{16}$. This is due to the abundance of individuals: incremental reasoning is based on locality-based modules [1], and these behave poorly in the presence of individuals. Aside from these two cases, the timings gathered using the incremental classifier were consistently below 5 seconds per version, across the corpus.

5 Culprit Finding

Upon completing the reasoner performance test we noted that, from $\mathcal{O}_{79}$ to $\mathcal{O}_{80}$ (in Figure 2), there is a significant performance improvement in HermiT. While our initial premise was to categorise logical diff-based impact between ontologies, now we encounter another problem: identifying and dissecting performance impact. We ascertained that the source of the bad performance is in the diff removals between those versions ($R = \mathrm{Removals}(\mathcal{O}_{79}, \mathcal{O}_{80})$), as with the additions of $\mathcal{O}_{80}$ the reasoning time was substantially lower. In order to investigate this phenomenon, we started with a brute-force *culprit finding* approach: for each axiom $\alpha \in R$ check if $\mathrm{RT}(\mathcal{O}_{80} \cup \{\alpha\}) \gg \mathrm{RT}(\mathcal{O}_{80})$. The size of R is 4,583 axioms, making this an expensive approach. It is also naive in the sense that culprits are not necessarily singleton sets. Nevertheless we examined $\mathrm{RT}(\mathcal{O}_{80} \cup \{\alpha \in R\})$ and found 13 (effectual) axioms which yield reasoning times ranging from 76 to 8,490 seconds. Surprisingly adding all

[9] HermiT returns a "StackOverflowError" when classifying $\mathcal{O}_{16}$, both in Protégé and OWL API.

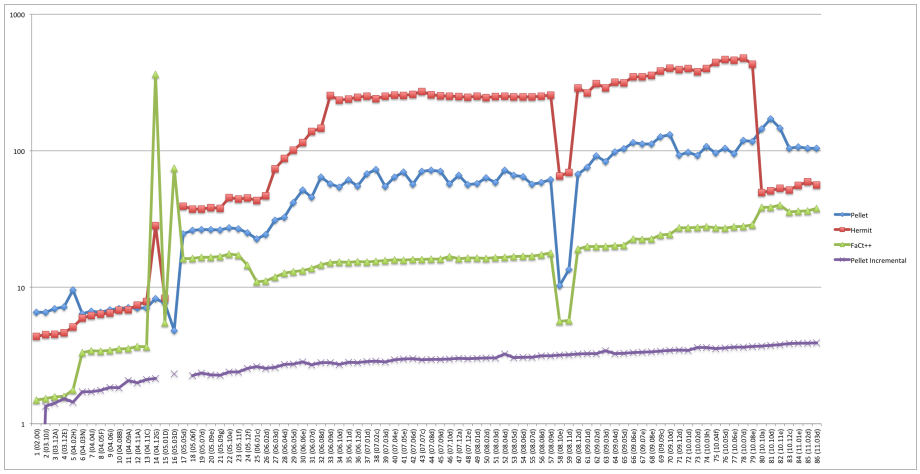

Fig. 2. Reasoner performance across NCIt (in seconds).

13 axioms to $\mathcal{O}_{80}$ results in a reasoning time of little over 9 hours. Thus some of the non-culprit additions exhibit a protective effect.

However, this approach is not only computationally expensive, but also relies on the existence of a diff which is not always available. We might want, given an "unmanageable" ontology, to find a subset thereof with which one can work with. As such we carried out a test partly based on the method described in [14], wherein we test the satisfiability checking time of each concept in the ontology. Such a test may be suggestive of the amount of time the reasoner spends on those concepts during classification (our culprit finding method is described in Algorithm 1). In order to extract a logically coherent subset of the ontology, which would be useful for repairing the culprit, we use the notion of a locality-based module [1]. We found a total of 12 concepts which have satisfiability checking times far greater than the average (see Table 2). The locality-based modules for the signature of the usage closure of each concept are significantly smaller than $\mathcal{O}_{79}$, the largest of which has 4,305 axioms (out of 116,587 logical axioms in $\mathcal{O}_{79}$). We found 9 modules $\mathcal{M}_i$ for which $\mathrm{RT}(\mathcal{O}_{79} \setminus \mathcal{M}_i)$ is nearly an order of magnitude faster than $\mathrm{RT}(\mathcal{O}_{79})$ ($\mathrm{RT}(\mathcal{O}_{79})$ = 430 seconds).

6 Discussion and Outlook

We have demonstrated with the diachronic study of the NCIt that merely syntactic diffs do not provide nearly enough insight into the impact of changes carried out, since logical differences are not identified. We found that ineffectual changes exist and account for a significant amount of logical changes throughout the NCIt. Such changes are discarded by semantic diffs, yet we show that they may provide helpful modelling insights. The axiom categorisation we devised allows ontology engineers to understand the lack

Algorithm 1 Identify subsets of an ontology $\mathcal{O}$ for which reasoning times are considerably better than the original ontology.

Input: Ontology $\mathcal{O}$
Output: Set of modules S, wherein for each $\mathcal{M}_i \in S$: $\mathrm{RT}(\mathcal{O} \setminus \mathcal{M}_i) \ll \mathrm{RT}(\mathcal{O})$

$S \leftarrow \emptyset$; $BadConcepts \leftarrow \emptyset$
for all concepts $C \in \widetilde{\mathcal{O}}$ **do**
  $Times \leftarrow Times \cup \langle C, SATtime(C) \rangle$
end for
for all $C \in \widetilde{\mathcal{O}}$ **do**
  if $SATtime(C) \geq$ average($SATtimes \in Times$) $\times$ 50 **then**
    $BadConcepts \leftarrow BadConcepts \cup C$
  end if
end for
for all $C \in BadConcepts$ **do**
  $\Sigma = \{$terms $t \in$ Usage(C)$\}$
  $\mathcal{M} = \top\bot$*-$mod(\Sigma)$
  if $\mathrm{RT}(\mathcal{O} \setminus \mathcal{M}) \ll \mathrm{RT}(\mathcal{O})$ **then**
    $S \leftarrow S \cup \mathcal{M}$
  end if
end for
return S

Concept	#$\mathcal{M}_i$	HermiT-RT($\mathcal{O} \setminus \mathcal{M}_i$)	Pellet-RT($\mathcal{O} \setminus \mathcal{M}_i$)
Cerebral_Glioblastoma	3029	56.7	38.1
TP53_Gene	3933	50.4	61.5
TP53_wt_Allele	3871	51.8	44.3
Erlotinib_Paclitaxel_Trastuzumab	4021	53.9	94.6
Tumor_Protein-p53	3894	51.4	102.2
Platelet-Derived_Growth_Factor_ Receptor-Like_Protein	3201	54.8	60.6
HRAS_wt_Allele	3302	63.1	44.2
p21_H-Ras_Protein	3329	62.9	89.7
AC-T-T_Regimen	4305	50.7	97.2

Table 2. Extracted culprits and corresponding concepts found in $\mathcal{O}$ (time in seconds).

of impact of their changes, and possibly refine these before publishing newer versions, particularly if redundancies are present.

From our structural analysis, we were able to gain considerable insight into the NCIt and its evolution. By looking at the entire history, it became relatively straightforward to identify tool artefacts and significant events and thus to disentangle accidental and essential features of the ontology. We are currently confirming our interpretation of various events with the EVS and thus far it conforms to their understanding of the history. Such an analysis is proving useful to the EVS as they find instances of the OWL version that do not correspond with their intent, and thus allowing them to publish corrections.

In the future we plan to apply a similar categorization to logically effectual changes. We also intend to examine the stability of entailments, i.e., whether an entailment persists throughout some or all NCIt versions. Finally, more elaborate forms of structural analysis, such as examining the justificatory structure [9], hold great promise for exposing the axiomatic richness of the modelling.

The reasoner performance results identify areas of performance weakness that would not have been evident using standard "grab a version" methods. Furthermore, we demonstrate the advantage (in terms of time) of using incremental reasoning for ontology engineering tasks, especially when large and complex ontologies are involved. We found in the NCIt corpus a realistic case for performance impact analysis, based on which we identified a number of meaningful culprits. The preliminary culprit finding methods and results described indicate that this approach works reasonably well. However the question of how to present these culprits to, and validate our approach with users still remains.

References

1. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. of Artificial Intelligence Research 31 (2008)
2. Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U.: OWL 2: The next step for OWL. J. of Web Semantics (2008)
3. Franconi, E., Meyer, T., Varzinczak, I.: Semantic diff as the basis for knowledge base versioning. In: Proc. of NMR-10 (2010)
4. Gatens, W., Konev, B., Ludwig, M., Wolter, F.: Versioning based on logical difference for lightweight description logic terminologies. In: Proc. of ARCOE-11 (2011)
5. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. of KR-06 (2006)
6. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga Llavori, R.: Building ontologies collaboratively using ContentCVS. In: Proc. of DL 2009. CEUR (`http://ceur-ws.org/`), vol. 477 (2009)
7. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga Llavori, R.: Supporting concurrent ontology development: Framework, algorithms and tool. Data and Knowledge Engineering 70(1) (2011)
8. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca Grau, B., Hendler, J.: Swoop: A Web ontology editing browser. J. of Web Semantics 4(2) (2006)
9. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: Proc. of ISWC/ASWC (2007)
10. Konev, B., Lutz, C., Walther, D., Wolter, F.: Logical difference and module extraction with CEX and MEX. In: Proc. of DL 2008. CEUR (`http://ceur-ws.org/`), vol. 353 (2008)
11. Křemen, P., Abrahamčík, J., Pufler, J., Šmíd, M.: OWLDiff (2008), `http://krizik.felk.cvut.cz/km/owldiff/`
12. Noy, N.F., Musen, M.A.: PROMPTDIFF: A fixed-point algorithm for comparing ontology versions. In: Proc. of AAAI-02 (2002)
13. W3C OWL Working Group: OWL 2 Web Ontology Language: Document overview. W3C Recommendation (27 Oct 2009), `http://www.w3.org/TR/owl2-syntax/`
14. Wang, T.D., Parsia, B.: Ontology performance profiling and model examination: First steps. In: Proc. of ISWC/ASWC-07. LNCS, vol. 4825. Springer-Verlag (2007)