Ontology Integration Using Mappings: Towards Getting the Right Logical Consequences

E. Jiménez-Ruiz[1], B. Cuenca Grau[2], I. Horrocks[2], and R. Berlanga[1]

[1] Universitat Jaume I, Spain, {ejimenez,berlanga}@uji.es
[2] University of Oxford, UK, {berg,ian.horrocks}@comlab.ox.ac.uk

Abstract. We propose a general method and novel algorithmic techniques to facilitate the integration of independently developed ontologies using mappings. Our method and techniques aim at helping users understand and evaluate the semantic consequences of the integration, as well as to detect and fix potential errors. We also present ContentMap, a system that implements our approach, and a preliminary evaluation which suggests that our approach is both useful and feasible in practice.

1 Introduction

Effective ontology integration techniques are often needed both during ontology development (e.g., when an ontology being developed reuses one or more external ontologies), and when ontologies are used in conjunction with data (e.g. when integrating and querying data sources annotated using different ontologies).

When the ontologies to be integrated have been independently developed, their vocabularies will most likely diverge, either because they use different namespaces, or because they use different names or naming conventions to refer to their entities. As a consequence, these ontologies will most likely be unrelated from a logical point of view, even if they intuitively overlap.

For example, suppose that Peter wants to develop an ontology about juvenile forms of arthritis.[3] He finds on the Web two independently developed ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ with different vocabularies, which describe different types of arthritis and juvenile diseases respectively, and decides that an integration of these two ontologies would make a good starting point. Although largely independent, the two ontologies do overlap; for example, both describe a particular form of juvenile arthritis known as JRA (Juvenile Rheumatoid Arthritis), although they use different vocabulary in their descriptions.

As a first step in the integration of such independently developed ontologies it is usually necessary to establish appropriate correspondences (or *mappings*) between the terms used in the various ontologies. For example, Peter may need to establish that Rheumatoid_Arthritis in $\mathcal{O}_1$ has the same intended meaning as Rheum_Arthritis in $\mathcal{O}_2$. Ontologies are, however, often too large and complex for ontology developers to manually establish these correspondences. To address

[3] The development of an ontology describing types of juvenile arthritis is one of the goals of the Health-e-Child project (http://www.health-e-child.org/).

this problem, a number of sophisticated tools have been developed (see `http://www.ontologymatching.org/` for a comprehensive list), the purpose of which is to (semi-) automatically establish appropriate correspondences.

When reasoning with the ontologies to be integrated together with the automatically generated mappings, however, errors are likely to occur. There are two main causes for these errors. On the one hand, mappings suggested by automated tools are likely to include some errors. On the other hand, even if the correct mappings have been found, the ontologies may contain conflicting descriptions of the overlapping entities. These errors manifest themselves as unintended logical consequences (e.g., unsatisfiable concepts or unintended subsumptions), and they can be difficult to detect, understand and repair. However, existing tools provide little or no support for the user in trying to obtain the right logical consequences when integrating ontologies using mappings—as discussed in Section 5, research in this area is still at an early stage [1, 2].

In this paper, we present a new general method and novel algorithmic techniques designed to provide just such support. We also present ContentMap,[4] a system that implements our approach, and a preliminary evaluation which suggests that our approach is both useful and feasible in practice.

We will assume some basic knowledge of OWL and Description Logics, and refer to [3, 4] for comprehensive overviews. Moreover, due to lack of space, proofs and some details concerning the empirical evaluation are included in an online technical report [5].

2 Problems with Ontology Integration using Mappings

In this section, we discuss some of the problems that arise when integrating ontologies using mappings, and analyze the requirements for a suitable tool.

Let us consider the ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ from Section 1 about arthritis and juvenile diseases respectively. As already mentioned, $\mathcal{O}_1$ and $\mathcal{O}_2$ overlap conceptually in the description of different types of juvenile rheumatoid arthritis. The overlapping parts of $\mathcal{O}_1$ and $\mathcal{O}_2$ are shown in Table 1.

Suppose that Peter—the ontology developer—uses an automatic mapping tool to find correspondences between entities in $\mathcal{O}_1$ and $\mathcal{O}_2$. Mappings are often represented as a tuple $\langle id, e_1, e_2, n, \rho \rangle$ [6], where id is a unique identifier for the mapping, e_1, e_2 are entity names in the vocabulary of $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively, n is a numeric confidence measure between 0 and 1, and ρ is a relation between e_1 and e_2 (typically subsumption ($\sqsubseteq$), equivalence ($\equiv$), or disjointness ($\bot$)).

Table 2 contains the mappings obtained by Peter using the mapping tool OLA (see `http://ola.gforge.inria.fr/`). The prefixes '1:' and '2:' for the entity names refer to the namespaces (omitted in Table 1) of $\mathcal{O}_1$ and $\mathcal{O}_2$. Note that the mappings and confidence values suggested by different tools can vary enormously, so experience and/or experimentation may be needed in order to select the most suitable tool (or combination of tools) for the problem at hand.

[4] A logiC-based ONtology inTEgratioN Tool using MAPpings

Table 1. Example Ontologies

	Ontology $\mathcal{O}_1$
α_1	Juvenile_Arthritis $\sqsubseteq$ Systemic_Disease $\sqcap$ Rheumatoid_Arthritis
α_2	Multi_Joint_Disease $\sqsubseteq$ Disease $\sqcap \geqslant 5$ affects.Joint
α_3	Rheumatoid_Arthritis $\sqsubseteq$ Disease $\sqcap \exists$has_Factor.$\top$
α_4	Poly_Juvenile_Arthritis $\sqsubseteq$ Juvenile_Arthritis $\sqcap$ Multi_Joint_Disease
α_5	Oly_Juvenile_Arthritis $\sqsubseteq$ Juvenile_Arthritis $\sqcap \neg$Poly_Juvenile_Arthritis
α_6	Oly_Juvenile_Arthritis $\sqsubseteq \forall$has_Factor.Negative_Factor
α_7	Negative_Factor $\sqcap$ Positive_Factor $\sqsubseteq \bot$
	Ontology $\mathcal{O}_2$
β_1	Juv_Rheum_Arthritis $\sqsubseteq$ Rheum_Arthritis $\sqcap$ Juv_Disease
β_2	Rheum_Arthritis $\sqsubseteq$ Disease $\sqcap \exists$has_Rheum_Factor.$\top$
β_3	Poly_Juv_Rheum_Arthritis $\sqsubseteq$ Juv_Rheum_Arthritis $\sqcap = 3$ affects.Joint
β_4	Poly_Juv_Rheum_Arthritis $\sqsubseteq \forall$has_Rheum_Factor.Positive_Rheum_Factor
β_5	Negative_Rheum_Factor $\sqcap$ Positive_Rheum_Factor $\sqsubseteq \bot$

As shown in Table 2, some of the mappings are clearly erroneous and, under any reasonable semantics for the mappings, will lead to unintended logical consequences. Indeed, the mappings μ_{14} and μ_{15} entail the logical equivalence of two concepts that are disjoint in both $\mathcal{O}_1$ and $\mathcal{O}_2$ (see also intended mappings μ_9 and μ_{10}); also, the mapping μ_{12} will lead to unintended subsumptions since not all forms of rheumatoid arthritis affect only children. Furthermore, even if only intended mappings had been generated, errors would still have occurred due to conflicting descriptions of some of the entities that occur in both ontologies. For example, the descriptions of Polyarticular Juvenile Rheumatoid Arthritis (axioms α_4, α_2 in $\mathcal{O}_1$ and β_3 in $\mathcal{O}_2$) are contradictory: in $\mathcal{O}_1$ it is described as a disease that affects at least 5 different joints, whereas in $\mathcal{O}_2$ it is said to affect exactly 3 joints. The intended mapping μ_7 will reveal this contradiction.

This example suggests some requirements for a tool supporting ontology integration using mappings. A suitable tool should provide support for: *(i)* generating sets of mappings, either manually or by selecting one or more mapping algorithms and setting their parameters; *(ii)* editing sets of mappings and filtering them according to different criteria; *(iii)* reasoning with the ontologies to be integrated together with the relevant mappings; *(iv)* comparing the entailments holding before and after the integration and detecting possible unintended entailments; *(v)* suggesting possible ways to repair the identified errors.

3 Proposed Method, Algorithms and Tool Support

In this paper, we present a novel approach for integrating ontologies using mappings that addresses the requirements from Section 2. To this end, a formal

Table 2. Example Mappings

Intended Mappings
$\langle \mu_1, \mathsf{1{:}Joint}, \mathsf{2{:}Joint}, 1.0, (\equiv)\rangle$;
$\langle \mu_2, \mathsf{1{:}Disease},\ \mathsf{2{:}Disease},\ 1,\ (\equiv)\rangle$;
$\langle \mu_3, \mathsf{1{:}Disease}, \mathsf{2{:}Disease}, 1.0, (\equiv)\rangle$;
$\langle \mu_4, \mathsf{1{:}affects}, \mathsf{2{:}affects}, 0.87, (\equiv)\rangle$;
$\langle \mu_5, \mathsf{1{:}Rheumatoid\_Arthritis},\ \mathsf{2{:}Rheum\_Arthritis},\ 1.0,\ (\equiv)\rangle$;
$\langle \mu_6, \mathsf{2{:}Juv\_Disease}, \mathsf{1{:}Disease}, 0.63, (\sqsubseteq)\rangle$;
$\langle \mu_7, \mathsf{1{:}Poly\_Juvenive\_Arthritis}, \mathsf{2{:}Poly\_Juv\_Rheum\_Arthritis}, 0.7, (\equiv)\rangle$;
$\langle \mu_8, \mathsf{1{:}has\_Factor}, \mathsf{2{:}has\_Rheum\_Factor}, 0.7, (\equiv)\rangle$;
$\langle \mu_9, \mathsf{1{:}Positive\_Factor}, \mathsf{2{:}Positive\_Rheum\_Factor}, 0.75, (\equiv)\rangle$;
$\langle \mu_{10}, \mathsf{1{:}Negative\_Factor}, \mathsf{2{:}Negative\_Rheum\_Factor}, 0.75, (\equiv)\rangle$;
$\langle \mu_{11}, \mathsf{2{:}Juv\_Rheum\_Arthritis},\ \mathsf{1{:}Rheumatoid\_Arthritis},\ 0.5,\ (\sqsubseteq)\rangle$;
Erroneous Mappings
$\langle \mu_{12}, \mathsf{1{:}Rheumatoid\_Arthritis},\ \mathsf{2{:}Juv\_Rheum\_Arthritis},\ 0.5,\ (\sqsubseteq)\rangle$;
$\langle \mu_{13}, \mathsf{1{:}Disease}, \mathsf{2{:}Juv\_Disease}, 0.63, (\sqsubseteq)\rangle$;
$\langle \mu_{14}, \mathsf{1{:}Positive\_Factor}, \mathsf{2{:}Negative\_Rheum\_Factor}, 0.63, (\equiv)\rangle$;
$\langle \mu_{15}, \mathsf{2{:}Positive\_Factor}, \mathsf{1{:}Negative\_Rheum\_Factor}, 0.63, (\equiv)\rangle$;

representation of ontology mappings should be adopted. This is crucial, for example, to reason unambiguously with the mappings together with the ontologies to be integrated, and thus to address requirements *(iii)–(v)* in Section 2.

A number of specialised semantics for ontology mappings have been proposed in the literature (e.g. [6, 7]). In this paper, however, we have chosen to represent ontology mappings as OWL 2 axioms [4]. Such a representation seems semantically coherent, and allows us to reuse the extensive range of OWL algorithmic techniques and infrastructure that is currently available. The general approach we present here, however, could be adapted to alternative formal representations of mappings (see Section 5 for further discussion).

We assume from now on that a set of mappings is represented as an OWL 2 ontology $\mathcal{M}$, where mappings $\langle id, e_1, e_2, n, \rho\rangle$ are given as axioms of the form $\mathsf{SubClassOf}(e_1\ e_2)$, $\mathsf{EquivalentClasses}(e_1\ e_2)$, or $\mathsf{DisjointClasses}(e_1\ e_2)$, for ρ of the form $(\sqsubseteq)$, $(\equiv)$, or $(\bot)$ respectively, with id (the mapping id) and n (the confidence value) added as axiom annotations [4]. We denote with $\mathsf{conf}(\alpha)$ the confidence value with which an axiom α is annotated. Representing mappings in this way gives them a standard "crisp" semantics, with ids and confidence values being represented only as annotations and thus having no effect on entailments.

Our general approach is given in Table 3. It is an interactive process where some steps involve computations that tools should perform automatically, while others (marked with $(\star)$) may require manual intervention. It consists of four main parts: *i)* Computation of mappings (Steps 1–3); *ii)* Computation of new entailments (Steps 4–6); *iii)* Detection of errors (Step 7); *iv*) Repair of the identified errors (Steps 8–12). We next present our method and describe its realisation in ContentMap—a Protégé 4 plugin available for download (see [5]).

Table 3. Ontology Integration Method

Input: $\mathcal{O}_1, \mathcal{O}_2$: ontologies with $\mathsf{Sig}(\mathcal{O}_1) = \Sigma_1, \mathsf{Sig}(\mathcal{O}_2) = \Sigma_2$ and $\Sigma_1 \cap \Sigma_2 = \{\top, \bot\}$
Output: $\mathcal{O}_1', \mathcal{O}_2'$: modified ontologies
$\mathcal{M}'$: Mappings between Σ_1 and Σ_2

1: ($\star$) Select mapping algorithm $\mathsf{map}(\mathcal{O}_1, \mathcal{O}_2)$
2: ($\star$) Select $\mathcal{M} \subseteq \mathsf{map}(\mathcal{O}_1, \mathcal{O}_2)$
3: ($\star$) **if** $\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}$ satisfactory, **then return** $\mathcal{O}_1' := \mathcal{O}_1, \mathcal{O}_2' := \mathcal{O}_2, \mathcal{M}' := \mathcal{M}$
4: $\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$
5: ($\star$) Select approximation functions $\mathsf{diff}^{\approx}$ and $\mathsf{mdiff}^{\approx}$
6: Compute $\Lambda = \mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}_1, \mathcal{U}) \cup \mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}_2, \mathcal{U}) \cup \mathsf{mdiff}^{\approx}_{\Sigma_1, \Sigma_2}(\mathcal{M}, \mathcal{U})$
7: ($\star$) Select $\Im^+, \Im^- \subseteq \Lambda$
8: ($\star$) Select $\mathcal{O}^- \subseteq \mathcal{U}$
9: $\mathbb{P}$:= all minimal plans for $\mathcal{U}$ given $\Im^+$, $\Im^-$, and $\mathcal{O}^-$
10: ($\star$) **if** no satisfactory plan in $\mathbb{P}$, **then** come back to either Step 2 or Step 7
11: ($\star$) Select $\mathcal{P} \in \mathbb{P}$
12: **return** $\mathcal{O}_1' := \mathcal{O}_1 \setminus \mathcal{P}, \mathcal{O}_2' := \mathcal{O}_2 \setminus \mathcal{P}, \mathcal{M}' := \mathcal{M} \setminus \mathcal{P}$

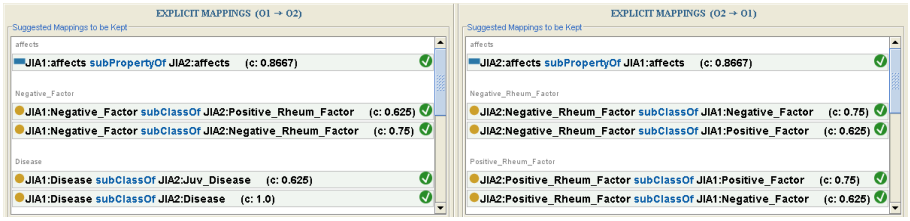


Fig. 1. GUI for Visualising Explicit Mappings in ContentMap

3.1 Computation of the Mappings

General Description Ontology mappings can be computed using one or more mapping tools (see Step 1), and can subsequently be refined (Step 2) in various ways (e.g., via manual selection or filtering according to a given threshold). After obtaining and possibly refining the mappings, a user may decide that the integration process is complete (Step 3), in which case it remains a purely syntactic process. They may, however, want to analyse the semantic consequences of the integration, in which case a reasoner is required.

Tool Support ContentMap provides for the input ontologies to be loaded, and for one or more mapping tools to be selected. Different weights can also be assigned to each of the mapping tools. It is also possible to load pre-computed mappings in the form of an OWL 2 ontology.

ContentMap provides a GUI for visualising the mappings (Fig. 1) and showing those that are to be accepted (marked with ✓) and those that are to be rejected (marked with ⊗). Users can automatically filter the mappings by setting a confidence threshold τ: mappings with a confidence value lower than τ are

automatically marked for rejection while those with a confidence value greater than τ are marked for acceptance. These selections can then be refined at will.

3.2 Computation of New Entailments

General Description To help users understand the semantic consequences of the integration, they should be informed about new entailments that hold in the merged ontology $\mathcal{U}$, but not in $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{M}$ alone. To this end, we use the notion of *deductive difference* [8, 9]. Intuitively, the deductive difference between $\mathcal{O}$ and $\mathcal{O}'$ w.r.t a signature Σ is the set of entailments constructed over Σ that do not hold in $\mathcal{O}$, but do hold in $\mathcal{O}'$.

Definition 1. *Let $\mathcal{DL}$ be a DL and Σ be a $\mathcal{DL}$-signature. The Σ-deductive difference between $\mathcal{DL}$-ontologies $\mathcal{O}$ and $\mathcal{O}'$ is given as follows:*

$$\mathsf{diff}_{\Sigma}(\mathcal{O},\mathcal{O}') = \{\alpha \mid \alpha \text{ a } \mathcal{DL}\text{-axiom, } \mathcal{O} \not\models \alpha,\ \mathcal{O}' \models \alpha \text{ and } \mathsf{Sig}(\alpha) \subseteq \Sigma\} \quad (1)$$

In our integration scenario, we want to inform the user about new entailments that hold in $\mathcal{O}_1$ or $\mathcal{O}_2$ as a result of the integration, i.e., $\mathsf{diff}_{\Sigma_1}(\mathcal{O}_1,\mathcal{U})$ and $\mathsf{diff}_{\Sigma_2}(\mathcal{O}_2,\mathcal{U})$. In our example from Tables 1 and 2, where the mappings are given as an OWL 2 ontology as described above, the entailments ($\mathsf{Positive\_Factor} \sqsubseteq \bot$) and ($\mathsf{Rheum\_Arthritis} \sqsubseteq \mathsf{Juv\_Disease}$) belong to $\mathsf{diff}_{\Sigma_1}(\mathcal{O}_1,\mathcal{U})$ and $\mathsf{diff}_{\Sigma_2}(\mathcal{O}_2,\mathcal{U})$ respectively. In addition, we want to inform the user about new entailments that contain symbols from both $\mathcal{O}_1$ and $\mathcal{O}_2$. These entailments can be seen as inferred mappings provided that the notion of a mapping is generalised to be an arbitrary DL axiom using terms from two signatures:

Definition 2. *Let $\mathcal{DL}$ be a DL and let Σ_1, Σ_2 be two $\mathcal{DL}$-signatures s.t. $\Sigma_1 \cap \Sigma_2 = \{\top, \bot\}$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$. A $\mathcal{DL}$-mapping between Σ_1 and Σ_2 is a $\mathcal{DL}$-axiom α over Σ such that $(\mathsf{Sig}(\alpha) \cap \Sigma_i) \setminus \{\top, \bot\} \neq \emptyset$ for each $i \in \{1, 2\}$. The axiom α is annotated with an id and with a confidence value $0 < \mathsf{conf}(\alpha) \leq 1$.*

Definition 1 can now be extended to take into account the new mappings, i.e.:

$$\mathsf{mdiff}_{\Sigma_1,\Sigma_2}(\mathcal{O},\mathcal{O}') = \{\alpha \in \mathsf{diff}_{\Sigma}(\mathcal{O},\mathcal{O}') \mid \alpha \ \mathcal{DL}\text{-mapping between } \Sigma_1,\ \Sigma_2\} \quad (2)$$

In our scenario, inferred mappings are captured by $\mathsf{mdiff}_{\Sigma_1,\Sigma_2}(\mathcal{M},\mathcal{U})$. For example, the mapping ($\mathsf{1{:}Juvenile\_Arthritis} \sqsubseteq \mathsf{2{:}Rheum\_Arthritis}$) would belong to this difference, because it is entailed by $\mathcal{U}$ but not by $\mathcal{M}$ alone.

The notion of deductive difference, however, has several drawbacks in practice. First, there is no algorithm for computing $\mathsf{diff}(\mathcal{O},\mathcal{O}')$ in expressive DLs such as $\mathcal{SROIQ}$ (OWL 2) and $\mathcal{SHOIQ}$ (OWL DL) [9]. For these languages, checking whether $\mathsf{diff}(\mathcal{O},\mathcal{O}') = \emptyset$ is undecidable. Currently, algorithms only exist for (fragments of) the OWL 2 EL and QL profiles [10, 9, 8]. Second, the number of entailments in the difference can be huge (even infinite), and so likely to overwhelm users. These drawbacks motivate the need for *approximations*—subsets of the difference—that the user can select in Step 5 from Table 3.

Definition 3. *A function $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O},\mathcal{O}')$ (resp. $\mathsf{mdiff}^{\approx}_{\Sigma_1,\Sigma_2}(\mathcal{O},\mathcal{O}')$) is an approximation for $\mathsf{diff}_{\Sigma}(\mathcal{O},\mathcal{O}')$ ($\mathsf{mdiff}_{\Sigma_1,\Sigma_2}(\mathcal{O},\mathcal{O}')$) if for any two $\mathcal{DL}$-ontologies $\mathcal{O},\mathcal{O}'$, $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O},\mathcal{O}') \subseteq \mathsf{diff}_{\Sigma}(\mathcal{O},\mathcal{O}')$ ($\mathsf{mdiff}^{\approx}_{\Sigma_1,\Sigma_2}(\mathcal{O},\mathcal{O}') \subseteq \mathsf{mdiff}_{\Sigma_1,\Sigma_2}(\mathcal{O},\mathcal{O}')$).*

Tool Support A useful approximation should be easy to compute, yet still provide meaningful information to the user. One possibility is to define an approximation by considering only entailments of a certain form.

The GUI implemented in ContentMap allows users to customise approximations by selecting among the following simple kinds of entailment, where A, B are atomic concepts (including $\top, \bot$) and R, S atomic roles or inverses of atomic roles: *(i)* $A \sqsubseteq B$, *(ii)* $A \sqsubseteq \neg B$, *(iii)* $A \sqsubseteq \exists R.B$, *(iv)* $A \sqsubseteq \forall R.B$, and *v)* $R \sqsubseteq S$. The smallest implemented approximation considers only axioms of the form *(i)*, while the largest one considers all types *(i)—(v)*.

All these approximations can be algorithmically computed, and ContentMap uses a number of optimisations for computing them. Clearly, the larger the class of entailments presented to the user, the more errors could be detected. The corresponding differences, however, are harder to compute, harder to present to the user, and may be harder for the user to understand.

3.3 Detection of Errors

General Description Some entailments in the relevant (approximate) differences may be intended, while others may reveal potential errors in the merged ontology $\mathcal{U}$. Step 7 therefore involves selecting entailments that: *(i)* are intended and should be entailed in $\mathcal{U}$ (written $\Im^+$ in Table 3), and *(ii)* are unintended and should not be entailed by $\mathcal{U}$ (written $\Im^-$).

In our example, the entailments (Positive_Factor $\sqsubseteq \bot$) and (Rheum_Arthritis $\sqsubseteq$ Juv_Disease) in $\mathsf{diff}_{\Sigma_1}(\mathcal{O}_1, \mathcal{U})$ and $\mathsf{diff}_{\Sigma_2}(\mathcal{O}_2, \mathcal{U})$ respectively are unintented and should belong to $\Im^-$. In contrast, the entailed mapping (1:Rheum_Arthritis $\sqsubseteq$ 2:Disease) is intended and should be included in $\Im^+$.

Tool Support The development of techniques to help users understand the relevant differences and subsequently select the sets of intended and unintended entailments is especially challenging.

First, the tool should explain why the new entailments that hold in $\mathcal{U}$ do not hold in $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{M}$ alone. The notion of a justification has proved very useful in understanding why an ontology entails a certain axiom [11, 12].

Definition 4. *Let $\mathcal{O} \models \alpha$. A justification for α in $\mathcal{O}$ is an ontology $\mathcal{O}' \subseteq \mathcal{O}$ satisfying the following properties:* (i) $\mathcal{O}' \models \alpha$, *and* (ii) *there is no $\mathcal{O}'' \subset \mathcal{O}'$ s.t. $\mathcal{O}'' \models \alpha$. We denote by* $\mathsf{Just}(\alpha, \mathcal{O})$ *the set of all justifications for α in $\mathcal{O}$.*

In order to explain a given entailment, ContentMap presents all its justifications. Computing justifications is expensive, so ContentMap uses the optimisations proposed in [13, 14] which have proved effective in practice.

Second, the potentially large number of relevant entailments may overwhelm the user. Hence, on the one hand, the tool should try to present these entailments in a way that makes them easier to understand and, on the other hand, it should provide suggestions about which entailments to include in $\Im^+$ and $\Im^-$.

Algorithm 1 Heuristic suggestions for $\Im^+$, $\Im^-$

Input: $\Lambda = \mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}_1, \mathcal{U}) \cup \mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}_2, \mathcal{U}) \cup \mathsf{mdiff}^{\approx}_{\Sigma_1,\Sigma_2}(\mathcal{M}, \mathcal{U})$ as in Table 3
τ_1, τ_2: real values with $0 < \tau_1 \leq \tau_2 \leq 1$
$\rhd$: dependency relation between axioms in Λ w.r.t. $\mathcal{U}$
Output: $\Im^+, \Im^-$: entailments suggested to be hold or not

```
1: ℑ+, ℑ− := ∅
2: for each α ∈ Λ do
3:     if α of the form A ⊑ ⊥ for A an atomic concept then
4:         ℑ− := ℑ− ∪ {α}
5:         for each β ∈ Λ such that α ▷* β do ℑ− := ℑ− ∪ {β}
6:     end if
7:     conf(α) := confidence of α as in Definition 7
8:     if conf(α) ≤ τ1 then ℑ− := ℑ− ∪ {α}
9:     if conf(α) ≥ τ2 and α ∉ ℑ− then ℑ+ := ℑ+ ∪ {α}
10: end for
11: return ℑ+, ℑ−
```

ContentMap exploits the dependencies between entailments to organise the way in which they are presented. Intuitively an entailment β depends on α in $\mathcal{O}$ if, whenever α is invalidated by removing a set of axioms from $\mathcal{O}$, then β is also invalidated. This implies that if $\alpha \in \Im^-$, then β should also be included in $\Im^-$ (and never in $\Im^+$). This idea can be formalised in our setting as follows:

Definition 5. *Let $\mathcal{O} \models \alpha, \beta$. The axiom β depends on α w.r.t. $\mathcal{O}$, written $\alpha \rhd \beta$, iff for each $\mathcal{J}_\beta \in \mathsf{Just}(\beta, \mathcal{O})$ there is $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O})$ s.t. $\mathcal{J}_\alpha \subseteq \mathcal{J}_\beta$.*

The relation $\rhd$ is consistent with our intuitions, as shown next:

Proposition 6. *Consider Table 3 and Definition 5. Let $\mathcal{U} \models \alpha, \beta$, $\mathcal{U}' \subset \mathcal{U}$ and $\alpha \rhd \beta$. Then: 1) $\mathcal{U}' \not\models \alpha$ implies $\mathcal{U}' \not\models \beta$, and 2) $\mathcal{U}' \models \beta$ implies $\mathcal{U}' \models \alpha$.*

In order to suggest to the user which entailments to include in $\Im^+$ and $\Im^-$, ContentMap exploits both the dependencies between entailments in the differences and the confidence values for each explicit mapping. We use the confidence values in the mappings to compute confidence values in the entailments from each of the obtained differences: we consider our confidence in an entailment to be equal to the maximum confidence we have in one of its justifications, and our confidence in a justification to be the product of our confidences in each of the axioms it contains. We formalise this in the following definition, in which we extend $\mathsf{conf}()$ to arbitrary (explicit) axioms and justifications. We assume that we have complete confidence in an axiom (i.e., $\mathsf{conf}(\alpha) = 1$) if it is not a mapping.

Definition 7 (Confidence in an Entailment). *Let $\mathcal{O}$ be an ontology, α an axiom in $\mathcal{O}$ that is not annotated with a confidence value, $\mathcal{J}$ a justification, and β an entailment s.t. $\mathcal{O} \models \beta$. We define $\mathsf{conf}(\alpha) = 1$, and $\mathsf{conf}(\mathcal{J})$ and $\mathsf{conf}(\beta)$ as follows:*

$$\mathsf{conf}(\mathcal{J}) = \prod_{\gamma \in \mathcal{J}} \mathsf{conf}(\gamma) \quad \textit{and} \quad \mathsf{conf}(\beta) = \mathsf{max}\Big(\bigcup_{\mathcal{J} \in \mathsf{Just}(\beta, \mathcal{O})} \mathsf{conf}(\mathcal{J}) \Big) \qquad (3)$$

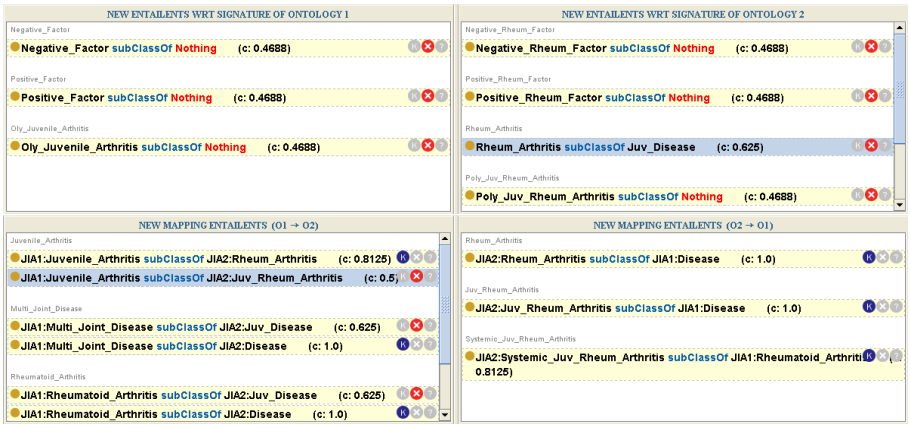


Fig. 2. GUI for Visualising New Entailments in ContentMap

The following proposition shows that the confidence values in Definition 7 are well-behaved w.r.t. Definition 5: if β depends on α then the confidence in β is smaller than the confidence in α.

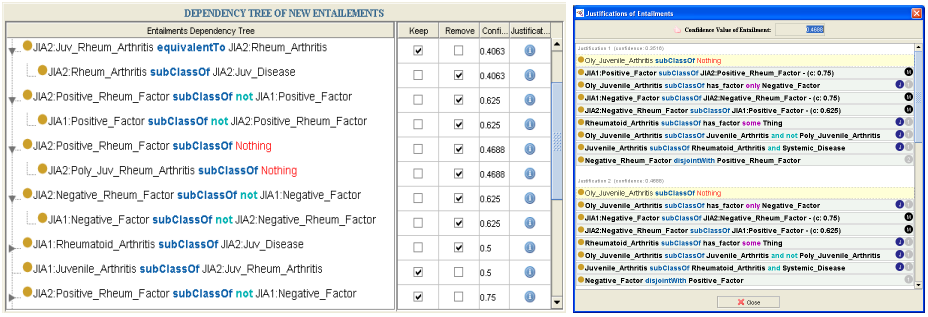
Proposition 8. *If* $\alpha \rhd^* \beta$, *then* $\mathsf{conf}(\beta) \leq \mathsf{conf}(\alpha)$

In order to compute initial suggestions for $\mathfrak{F}^-$ and $\mathfrak{F}^+$, ContentMap implements the heuristics in Algorithm 1, which are based on Definitions 5, 7 and Propositions 6, 8. The algorithm accepts as input the entailments in the relevant differences, their dependencies, and two confidence thresholds τ_1 and τ_2. For each input entailment α, Algorithm 1 uses $\rhd$, τ_1 and τ_2 to either include it in $\mathfrak{F}^+$ (i.e. α is intended), in $\mathfrak{F}^-$ (i.e. α is unintended) or in neither of them. An entailment α is included in $\mathfrak{F}^-$ if it reveals a contradiction (Line 3), depends on a contradiction (Line 5), or if its confidence according to Definition 7 is lower than τ_1 (Line 8). In contrast, α is included in $\mathfrak{F}^+$ if it is not contained in $\mathfrak{F}^-$ and its confidence is higher than the threshold τ_2 (Line 9).

Figures 2 and 3 show the ContentMap GUI for selecting $\mathfrak{F}^+$ and $\mathfrak{F}^-$. Figure 2 shows the new entailments in each of the computed differences. The left-hand side of Figure 3(a) displays the dependency relation as a hierarchy, which can be expanded and contracted in the usual way; on the right-hand side of Figure 3(a), the user can select an entailment to be in $\mathfrak{F}^+$ or $\mathfrak{F}^-$ and show its justifications (Figure 3(b)). When displaying justifications, the GUI indicates whether the axioms in these justifications come from $\mathcal{O}_1$, $\mathcal{O}_2$, or the mappings $\mathcal{M}$ (marked with '1', '2' and 'M' respectively). The entailments in Figures 2 and 3(a) already marked as intended/unintended are those suggested by Algorithm 1.

3.4 Repair

General Description If the user has selected one or more unintended entailments (i.e., $\mathfrak{F}^- \neq \emptyset$), then $\mathcal{U}$ clearly contains errors. These errors can always



(a) Dependency Relation (b) Justifications

Fig. 3. GUI for Visualising the Dependency Relation and the Justifications

be repaired by removing axioms from $\mathcal{U}$—in the limit, removing *all* the axioms from $\mathcal{U}$ will eliminate all entailments. However, any removal of axioms should also respect $\Im^+$, the user's selection of intended entailments, i.e., the entailments in $\Im^+$ should still hold after any removal of axioms.

As mentioned in Section 1, errors could be due to erroneous mappings, to inherently conflicting information in the two ontologies, or to some combination of both. Repairing such errors might, therefore, require the removal of axioms from $\mathcal{M}$, $\mathcal{O}_1$ and $\mathcal{O}_2$. The user may, however, have a (principled or pragmatic) preference regarding the source of axioms to be removed; e.g., $\mathcal{M}$ might be considered the most likely source of errors, or it may be impossible to change (one or both of) $\mathcal{O}_1$ and $\mathcal{O}_2$. We therefore define the notion of a *repair plan* (or *plan* for short) for an ontology $\mathcal{O}$ w.r.t. the (un-) intended entailments $\Im^-$ and $\Im^+$, and a subset $\mathcal{O}^-$ of $\mathcal{O}$, where the axioms in $\mathcal{O}^-$ are those that are allowed to be removed from $\mathcal{O}$. Such a plan is simply a subset of the axioms in $\mathcal{O}^-$ whose removal from $\mathcal{O}$ both eliminates the entailments in $\Im^-$ and preserves those in $\Im^+$. In general, there may be zero or more such plans.

Definition 9. *Let $\mathcal{O}$, $\Im^+$, $\Im^-$, and $\mathcal{O}^-$ be finite sets of axioms such that $\mathcal{O}^- \subseteq \mathcal{O}$, $\mathcal{O} \models \Im^-$, $\mathcal{O} \models \Im^+$, and $\Im^+ \cap \Im^- = \emptyset$.*

A repair plan for $\mathcal{O}$ given $\mathcal{O}^-$, $\Im^+$ and $\Im^-$ is a set $\mathcal{P} \subseteq \mathcal{O}^-$ such that: 1) *$(\mathcal{O} \setminus \mathcal{P}) \models \alpha$ for each $\alpha \in \Im^+$, and* 2) *$(\mathcal{O} \setminus \mathcal{P}) \not\models \beta$ for each $\beta \in \Im^-$. A plan $\mathcal{P}$ is minimal if there is no $\mathcal{P}_1$ such that $\mathcal{P}_1 \subset \mathcal{P}$.*

In our method, $\Im^-$, $\Im^+$ and $\mathcal{O}^-$ are selected by the user in Steps 7 and 8 from Table 3. In our example, if Peter imports $\mathcal{O}_1$ and $\mathcal{O}_2$ from the Web, and if he is not willing to copy and modify them (e.g., if he wants to always import the latest versions), then plans should only remove axioms from the mappings (i.e., $\mathcal{O}^- = \mathcal{M}$). However, the descriptions of Polyarticular Juvenile Rheumatoid Arthritis in $\mathcal{O}_1$ and $\mathcal{O}_2$ are inherently in contradiction and hence, if $\mathcal{O}^- = \mathcal{M}$, the corresponding error cannot be fixed without deleting reasonable mappings. In this case a more sensible choice would be to repair one or both of $\mathcal{O}_1$ and $\mathcal{O}_2$.

Algorithm 2 Computing All Plans Using Justifications

Procedure all_Plans($\mathcal{O}, \mathcal{O}^-, \Im^+, \Im^-$)
Input: $\mathcal{O}$, $\mathcal{O}^-$, $\Im^+$, $\Im^-$ as in Definition 9, Just($\gamma, \mathcal{O}$) for each $\gamma \in \Im^+ \cup \Im^-$
Output: **P**: set of all plans

```
1: P := ∅
2: for each P ⊆ O⁻ do
3:     validPlan := true
4:     for each α ∈ ℑ⁺ do
5:         foundJust := false
6:         for each J_α ∈ Just(α, O) do
7:             if J_α ⊆ O \ P then foundJust := true
8:         end for
9:         if foundJust = false then validPlan := false
10:    end for
11:    for each β ∈ ℑ⁻ and J_β ∈ Just(β, O) do
12:        if J_β ⊆ O \ P then validPlan := false
13:    end for
14:    if validPlan = true then P := P ∪ {P}
15: end for
16: return P
```

Tool Support Definition 9 suggests a simple procedure for computing all plans: for each possible $\mathcal{P} \subseteq \mathcal{O}^-$, use a reasoner to check if $\mathcal{P}$ is a valid plan, i.e., satisfies Conditions 1 and 2 from Definition 9. ContentMap, however, implements an algorithm (see Algorithm 2) that avoids potentially expensive entailment checks by reusing the justifications already computed when obtaining the dependency relation from Definition 5. The algorithm is based on the following principles: *(i)* in order for $\alpha \in \Im^+$ to hold after the execution of a plan $\mathcal{P}$, the ontology $\mathcal{O} \setminus \mathcal{P}$ must contain at least one justification for α in $\mathcal{O}$ (see Lines (6-8)); *(ii)* in order for $\beta \in \Im^-$ not to hold after the execution of $\mathcal{P}$, it is sufficient to show that no justification for β in $\mathcal{O}$ is in $\mathcal{O} \setminus \mathcal{P}$ (Lines 11-13). The set of all minimal plans can be straightforwardly computed once all the plans have been obtained.

Proposition 10. *Algorithm 2 computes all the plans for $\mathcal{O}$ given $\mathcal{O}^-$, $\Im^+$, $\Im^-$.*

Note that conflicting choices in $\Im^+$ and $\Im^-$ may make it impossible to find any plans. Some of these conflicts can be detected using the dependency relation $\rhd$, as shown in the following proposition:

Proposition 11. *Let $\mathcal{O}$, $\Im^+$, $\Im^-$ and $\mathcal{O}^-$ be as in Definition 9 and let $\mathcal{O} \models \alpha, \beta$. If $\alpha \rhd^* \beta$ w.r.t. $\mathcal{O}$, $\alpha \in \Im^-$ and $\beta \in \Im^+$, then no plan exists.*

There may also be particular selections of $\mathcal{O}^-$, $\Im^+$ and $\Im^-$ for which the number of minimal plans is very large. In this case, selecting the most suitable minimal plan becomes difficult for users.

In order to assist users in the selection of a minimal plan (Steps 10-11), ContentMap implements a number of heuristics based on the confidence of the mappings in the plans and the total number of axioms in the plan, among others.

4 Evaluation

In our experiments, we used a suite of four ontologies adapted from the 2004 EON Ontology Alignment Contest[5] which describe the domain of bibliographic references. These ontologies have been developed independently by INRIA ($\mathcal{O}_{\mathsf{INR}}$), MIT ($\mathcal{O}_{\mathsf{MIT}}$), UMBC ($\mathcal{O}_{\mathsf{UMBC}}$) and AIFB Karlsruhe ($\mathcal{O}_{\mathsf{AIFB}}$) respectively. Their sizes vary from 58 classes, 46 object properties, 26 data properties and 235 axioms in $\mathcal{O}_{\mathsf{AIFB}}$ to 18 classes, 12 object properties, 19 data properties and 96 axioms in $\mathcal{O}_{\mathsf{UMBC}}$. Even if small, all these ontologies are fairly expressive ($\mathcal{O}_{\mathsf{INR}}$ and $\mathcal{O}_{\mathsf{MIT}}$ are expressible in the DL $\mathcal{ALCHQ}(\mathcal{D})$, $\mathcal{O}_{\mathsf{UMBC}}$ in $\mathcal{ALCIN}(\mathcal{D})$ and $\mathcal{O}_{\mathsf{AIFB}}$ in $\mathcal{ALCI}(\mathcal{D})$ respectively). Their average classification time is less than a second when using Pellet 1.5.

In the 2004 EON contest, $\mathcal{O}_{\mathsf{INR}}$ was used as reference ontology, and the other ontologies were independently aligned with it using each of the competing tools. For evaluation, a manually produced gold standard containing the agreed-upon, correct mappings was provided for each pair of aligned ontologies. The gold standards for $\mathcal{O}_{\mathsf{MIT}}$, $\mathcal{O}_{\mathsf{UMBC}}$ and $\mathcal{O}_{\mathsf{AIFB}}$ contain 119, 83 and 98 mappings respectively. All the test ontologies and gold standards are available at our web page [5].

The experiments are organised in two parts, which we specify next, and were performed on a laptop with a 1.82 GHz processor and 3GB of RAM.

Repair of Ontologies Using Gold Standard First, we have evaluated the semantic consequences of integrating each of $\mathcal{O}_{\mathsf{MIT}}$, $\mathcal{O}_{\mathsf{UMBC}}$ and $\mathcal{O}_{\mathsf{AIFB}}$ with $\mathcal{O}_{\mathsf{INR}}$ using the corresponding gold standard to detect inherent disagreements between them. In each case we have applied our method from Table 3, with the mappings in Step 2 being the corresponding gold standard. We have used both the smallest and the largest approximations of the deductive differences implemented in ContentMap (see Section 3.2), and obtained the following results.

1. Alignment $\mathcal{O}_{\mathsf{MIT}}$, $\mathcal{O}_{\mathsf{INR}}$: for the smallest (resp. the largest) approximation, there were 3 (resp. 33) new entailments in $\mathcal{O}_{\mathsf{MIT}}$, 13 (resp. 85) in $\mathcal{O}_{\mathsf{INR}}$ and 35 (resp. 189) new mappings.
2. Alignment $\mathcal{O}_{\mathsf{UMBC}}$, $\mathcal{O}_{\mathsf{INR}}$: for the smallest (largest) approximation, there were 2 (7) new entailments in $\mathcal{O}_{\mathsf{UMBC}}$, 10 (300) in $\mathcal{O}_{\mathsf{INR}}$ and 28 (176) new mappings.
3. Alignment $\mathcal{O}_{\mathsf{AIFB}}$, $\mathcal{O}_{\mathsf{INR}}$:for the smallest (largest) approximation, there were 4 (37) new entailments in $\mathcal{O}_{\mathsf{AIFB}}$, 2 (19) in $\mathcal{O}_{\mathsf{INR}}$ and 46 (140) new mappings.

Note that the number of new entailments (and hence the amount of information that ContentMap shows to users) largely depends on the selected approximation.

In all cases we found a significant number of obviously unintended inferences due to inherent disagreements between the ontologies being integrated. For example, when integrating $\mathcal{O}_{\mathsf{AIFB}}$ and $\mathcal{O}_{\mathsf{INR}}$, our tool detected a total of 34 newly unsatisfiable concepts in the two ontologies. The large number of (rather complex) justifications for some of these entailments would make manual repair

[5] `http://oaei.ontologymatching.org/2004/Contest/`

Table 4. Synthetic Experiments

Input: $\mathcal{O}_1, \mathcal{O}_2$: ontologies with $\mathsf{Sig}(\mathcal{O}_1) = \Sigma_1, \mathsf{Sig}(\mathcal{O}_2) = \Sigma_2$ and $\Sigma_1 \cap \Sigma_2 = \{\top, \bot\}$
$\mathcal{M}$: automatically-generated mappings between Σ_1 and Σ_2, $\mathcal{M}_\mathcal{G}$: gold standard

1: Filter $\mathcal{M}$ given a confidence threshold τ
2: Compute recall and precision of $\mathcal{M}$ with respect to $\mathcal{M}_\mathcal{G}$
3: Compute $\mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}_1, \mathcal{U})$, $\mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}_2, \mathcal{U})$ and $\mathsf{mdiff}^{\approx}_{\Sigma_1,\Sigma_2}(\mathcal{M}, \mathcal{U})$ for $\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$
4: Compute all minimal plans for $\mathcal{U}$ given $\mathcal{O}^- = \mathcal{M}$, $\Im^+ = \emptyset$, and $\Im^-$:= new concept unsatisfiability entailments.
5: Compute $\mathcal{O}'_1 := \mathcal{O}_1 \setminus \mathcal{P}, \mathcal{O}'_2 := \mathcal{O}_2 \setminus \mathcal{P}, \mathcal{M}' := \mathcal{M} \setminus \mathcal{P}$, for $\mathcal{P}$ the best plan.
6: Compute recall and precision of $\mathcal{M}'$ with respect to $\mathcal{M}_\mathcal{G}$
7: Compute $\Lambda = \mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}'_1, \mathcal{U}') \cup \mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}'_2, \mathcal{U}') \cup \mathsf{mdiff}^{\approx}_{\Sigma_1,\Sigma_2}(\mathcal{M}', \mathcal{U}')$ for $\mathcal{U}' := \mathcal{O}'_1 \cup \mathcal{O}'_2 \cup \mathcal{M}'$; and Compute dependency relation $\rhd$ over Λ
8: Compute all minimal plans for $\mathcal{U}'$ given $\mathcal{O}^- = \mathcal{M}'$, $\Im^+ = \emptyset$ and $\Im^-$:= as obtained in **Algorithm 1** for a pre-fixed confidence threshold τ_1.
9: Compute $\mathcal{O}''_1 := \mathcal{O}'_1 \setminus \mathcal{P}, \mathcal{O}''_2 := \mathcal{O}'_2 \setminus \mathcal{P}, \mathcal{M}'' := \mathcal{M}' \setminus \mathcal{P}$, for $\mathcal{P}$ the best plan.
10: Compute recall and precision of $\mathcal{M}''$ with respect to $\mathcal{M}_\mathcal{G}$

extremely challenging. When computing the corresponding plans, we found that the problem originated in the incompatible ranges of two datatype properties; hence, the modification of two axioms was sufficient to eliminate all the errors. In most cases, a significant number of obviously unintended new subsumptions were also detected. For example, merging $\mathcal{O}_{\mathsf{MIT}}$ and $\mathcal{O}_{\mathsf{INR}}$ resulted in the new subsumptions $\mathsf{TechnicalReport} \sqsubseteq \mathsf{Date}$ and $\mathsf{TechnicalReport} \sqsubseteq \exists \mathsf{date.Reference}$. Again, the obtained plans revealed the origin of the problems and facilitated their repair.

Synthetic Repair of Automatically Generated Mappings Having repaired the test ontologies using the gold standard mappings, we used ContentMap to automatically detect and repair errors resulting from the generation of new mappings using the mapping tools OLA , AROMA[6] and CIDER[7]. Our goal was twofold: first, to show that the algorithms in ContentMap are practical; second, to show that ContentMap can be used to automatically detect and repair errors, as well as to improve the quality of automatically generated mappings.

For each pair of test ontologies, and for various sets of automatically generated mappings, we performed the synthetic experiments described in Table 4, which closely follow our proposed method from Table 3. In contrast to Table 3, however, the repair of errors is performed in two stages: first, the obvious errors (i.e. unsatisfiable concepts); then, those subsumptions that ContentMap heuristically found unintended. For these experiments we have used the smallest approximation of the deductive difference available in ContentMap.

Due to space limitations, here we only summarise the main conclusions; the complete results are, however, available in our technical report [5]. First, from a

[6] AROMA: http://www.inrialpes.fr/exmo/people/jdavid/
[7] CIDER: http://sid.cps.unizar.es/SEMANTICWEB/ALIGNMENT/

computational point of view, the main bottleneck is the computation of all the justifications for the entailments of interest. Once the justifications have been computed, the time needed for computing the plans is relatively short. Hence, it is important to investigate optimisations for computing all justifications; first steps in this direction have been taken in [13, 14].

Second, the number of entailments in the computed differences is similar to the results obtained using the gold standard with the same approximation, and the number of obtained minimal plans varies from 1-50. Furthermore, the use of the dependencies given in Definition 5 significantly reduces the amount of information that the user would need to consider.

Third, the use of automatically generated mappings did result in the occurrence of a significant number of unintended entailments. For example, when aligning $\mathcal{O}_{\mathsf{AIFB}}$ and $\mathcal{O}_{\mathsf{INR}}$ using CIDER with confidence threshold $\tau = 0.1$, we found 55 new unsatisfiable concepts. After fixing these errors, ContentMap found 34 unintended subsumption relationships using Algorithm 1 with a threshold $\tau_1 = 0.3$. Moreover, we checked manually selected examples and found out that the heuristically detected unintended subsumptions were indeed errors; furthermore, these errors were mainly caused by incorrect mappings.

Fourth, the application of the plans resulted in the automatic correction of the identified errors, which resulted in an improvement in the precision of the automatically generated mappings. The improvement in precision occurred in all cases and varied from 1%-5%, achieving a good balance with respect to the recall, which remained unchanged in most cases, although a decrease from 1%-3% was observed in a few isolated cases.

5 Related Work and Conclusion

In the last few years, the problem of automatically generating mappings between ontologies has been extensively investigated. An comprehensive and up-to-date source of information about the topic (including papers, tools, ontologies for evaluation, etc.) can be found at `http://www.ontologymatching.org/`.

The problem of reasoning with the generated mappings has also received significant attention. Different semantics have been proposed for assigning formal meaning to ontology mappings (e.g. [6, 7]). As mentioned in Section 3, our general approach is applicable regardless of the semantics adopted for the mappings.

Recently, research has been conducted on the debugging and revision of mappings [1, 2]. This research has so far been focused on mappings represented using Distributed Description Logics (DDL) [7]. Our work is also related to the existing approaches for debugging and repairing inconsistencies in OWL ontologies (e.g. [15, 12, 11, 13]). In both of these lines of research, the detected and repaired errors are limited to unsatisfiable concepts and inconsistent ontologies.

When compared to this existing work, we believe that our approach presents a number of improvements. First,the entailments to be repaired are not restricted to obvious inconsistencies, but can include any unintended entailment. Second, users can customise the kinds of entailments to be taken into account when

comparing the integrated ontology to the individual ones in order to detect errors (i.e., select the approximation of the deductive difference). Third, users can select not only which entailments should be invalidated, but also which ones should necessarily hold upon completion of the repair process. To this end, we provide a number of novel techniques for helping the user to select which entailments are (un)intended, such as the computation of the dependencies between entailments (the relation $\rhd$), confidence in entailments according to the confidence in the mappings, etc. Fourth, we provide efficient algorithms for computing *all* the repair plans and to help the user select the most suitable plan from amongst those computed. Fifth, when compared to existing work on debugging and repair in OWL, we provide a clear distinction between the ontologies being integrated and the mappings, and users can customise the ontologies from which the plans are allowed to delete axioms. Finally, we provide a fully-fledged editor and reasoning infrastructure integrated with Protégé 4.

References

[1] Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Reasoning Support for Mapping Revision. Journal of Logic and Computation (2008)

[2] Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Repairing ontology mappings. In: Proc. of AAAI. (2007) 1408–1413

[3] Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From $\mathcal{SHIQ}$ and RDF to OWL: the making of a web ontology language. J. Web Sem. **1**(1) (2003) 7–26

[4] Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U.: OWL 2: The next step for OWL. J. Web Semantics **6**(4) (2008) 309–322

[5] Jimenez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Ontology integration using mappings: Towards getting the right logical consequences. Technical report (2008) Available at http://krono.act.uji.es/people/Ernesto/contentmap. The ContentMap tool is also available at this location.

[6] Euzenat, J.: Semantic precision and recall for ontology alignment evaluation. In: Proc. of IJCAI. (2007) 348–353

[7] Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. J. Data Semantics **1** (2003) 153–184

[8] Kontchakov, R., Wolter, F., Zakharyaschev, M.: Can you tell the difference between DL-Lite ontologies? In: Proc. of KR. (2008)

[9] Konev, B., Walther, D., Wolter, F.: The logical difference problem for description logic terminologies. In: Proc. of IJCAR. (2008) 259–274

[10] Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language: Profiles. W3C Working Draft (2008)

[11] Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A.: Debugging unsatisfiable classes in OWL ontologies. J. Web Semantics **3**(4) (2005) 268–293

[12] Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. J. Autom. Reasoning **39**(3) (2007) 317–349

[13] Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: Proc. of ISWC. (2007) 267–280

[14] Suntisrivaraporn, B., Qi, G., Ji, Q., Haase, P.: A modularization-based approach to finding all justifications for OWL DL entailments. In: Proc. of ASWC. (2008)

[15] Kalyanpur, A., Parsia, B., Sirin, E., Grau, B.C.: Repairing unsatisfiable concepts in OWL ontologies. In: Proc. of ESWC. (2006) 170–184