A Novel Approach and Practical Algorithms for Ontology Integration

Giorgos Stoilos(✉), David Geleta, Jetendr Shamdasani, and Mohammad Khodadadi

Babylon Health, London, SW3 3DD, UK
{giorgos.stoilos,david.geleta,jetendr.shamdasani, mohammad.khodadadi}@babylonhealth.com

Abstract. Today a wealth of knowledge and data are distributed using Semantic Web standards. Especially in the (bio)medical domain several sources like SNOMED, NCI, FMA, and more are distributed in the form of OWL ontologies. These can be matched and integrated in order to create one large medical Knowledge Base. However, an important issue is that the structure of these ontologies may be profoundly different hence using the mappings as initially computed can lead to incoherences or changes in their original structure which may affect applications. In this paper we present a framework and novel approach for integrating independently developed ontologies. Starting from an initial seed ontology which may already be in use by an application, new sources are used to iteratively enrich and extend the seed one. To deal with structural incompatibilities we present a novel fine-grained approach which is based on mapping repair and alignment conservativity, formalise it and provide an exact as well as approximate but practical algorithms. Our framework has already been used to integrate a number of medical ontologies and support real-world healthcare services provided by Babylon Health. Finally, we also perform an experimental evaluation and compare with state-of-the-art ontology integration systems that take into account the structure and coherency of the integrated ontologies obtaining encouraging results.

Keywords: Ontology integration · Ontology matching
Ontology alignment · Conservativity · Mapping repair

1 Introduction

Today a wealth of knowledge and data are distributed using Semantic Web technologies and standards. For example, the Linked Open Vocabularies effort [22] contains more than 600 ontologies for various subjects like geography, multimedia, security, geometry, and more. Especially in the biomedical domain, a large number of ontologies have been developed during the previous decades like SNOMED,[1] NCI [5], UMLS,[2] the Disease ontology [16] and many more, while

[1] https://www.snomed.org/.

[2] https://uts.nlm.nih.gov/home.html.

D. Vrandečić et al. (Eds.): ISWC 2018, LNCS 11136, pp. 458–476, 2018.
https://doi.org/10.1007/978-3-030-00671-6_27

```
Algorithm 1. postProcessKBStructure($\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}$, Config)
Input: Two coherent ontologies $\mathcal{O}_1, \mathcal{O}_2$ and a set of mappings $\mathcal{M}$ between them.
$\mathcal{M}_{\text{m-1}} := \{\langle C_i, D\rangle \mid \{\langle C_i, D\rangle, \langle C_j, D\rangle\} \subseteq \mathcal{M} \wedge C_i \neq C_j\}$.
$\mathcal{M}' := \mathcal{M} \setminus \mathcal{M}_{\text{m-1}}$
for all $D \in \mathsf{Sig}(\mathcal{O}_2)$ do
    $\mathcal{M}' := \mathcal{M}' \cup \mathsf{disambiguate\text{-}m\text{-}1}(\{\langle C_i, D\rangle \mid \langle C_i, D\rangle \in \mathcal{M}_{\text{m-1}}\}, \mathsf{Config})$
end for
$\mathsf{Exclusions} := \emptyset$
$\mathsf{ConflictSets} := \{\{m_1, m_2\} \mid \mathcal{O}_1 \cup \mathcal{O}_2 \cup \{m_1, m_2\} \models A \sqsubseteq B, \mathcal{O}_1 \not\models A \sqsubseteq B\}$
for all $\{\langle A, A'\rangle, \langle B, B'\rangle\} \in \mathsf{ConflictSets}$ with $\mathcal{O}_2 \models_{\mathsf{rdfs}} A' \sqsubseteq B'$ do
    $\mathsf{Exclusions} := \mathsf{Exclusions} \cup \{A' \sqsubseteq E \mid A' \sqsubseteq E \in \mathcal{O}_2, \mathcal{O}_2 \models_{\mathsf{rdfs}} E \sqsubseteq B'\}$
end for
return $\langle \mathcal{O}_2 \setminus \mathsf{Exclusions}, \mathcal{M}'\rangle$
```

BioPortal [15] is a repository of more than 600 biomedical ontologies. Identifying the common entities between these vocabularies and integrating them is beneficial for building ontology-based applications as one could unify complementary information that these vocabularies contain building a "complete" Knowledge Base (KB).

The problem of computing correspondences (mappings) between different ontologies is referred to as *ontology matching* or *alignment* [18]. A number of ontology matching systems have been developed in the past and have shown to behave well in practice. However, besides classes with their respective labels ontologies usually bring a class hierarchy and depending on how they have been conceptualised they may exhibit significant incompatibilities. For example, in NCI proteins are declared to be disjoint from anatomical structures whereas in FMA proteins are subclasses of anatomical structures. Hence, integrating them without taking into account their logical axioms may lead to many undesired consequences like unsatisfiable classes [11] and/or changes in their initial structure [6]. For these reasons the notions of *conservative alignment* [6,19,20] and *mapping repair* [8,12] have been proposed in the literature. These notions dictate that the mappings should not alter the original ontology structure or introduce unsatisfiable concepts. If they do, then a so-called *violation* occurs which needs to be *repaired* by discarding some of the mappings.

Unfortunately, dropping mappings introduces another problem which is the increase of ambiguity and redundancy. For example, if one drops all mappings between NCI and FMA proteins (due to their structural incompatibilities), then the integrated ontology will contain at least two classes for the same real-world entity. Apart from an unnecessary increase in the size of the integrated ontology this introduces ambiguity and decreases interoperability between services that use classes from the KB. The problem becomes more acute if further sources are integrated in which case we may end up with multiple classes representing the same real-world entity.

Algorithm 2. $\mathsf{postProcessNewOntoStructure}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}, \mathsf{Config})$

```
Input: Two ontologies $\mathcal{O}_1, \mathcal{O}_2$ and a set of mappings $\mathcal{M}$ computed between them.
$\mathcal{M}_{\text{1-m}} := \{\langle C, D_i\rangle \mid \{\langle C, D_i\rangle, \langle C, D_j\rangle\} \subseteq \mathcal{M} \wedge D_i \neq D_j\}$.
$\mathcal{M}' := \mathcal{M} \setminus \mathcal{M}_{\text{1-m}}$
for all $C \in \mathsf{Sig}(\mathcal{O}_1)$ do
    $\mathcal{M}' := \mathcal{M}' \cup \mathsf{disambiguate\text{-}1\text{-}m}(\{\langle C, D_i\rangle \mid \langle C, D_i\rangle \in \mathcal{M}_{\text{1-m}}\}, \mathsf{Config})$
end for
$\mathsf{ConflictSets} := \{\{m_1, m_2\} \mid \mathcal{O}_1 \cup \mathcal{O}_2 \cup \{m_1, m_2\} \models A \sqsubseteq B, \mathcal{O}_2 \not\models A \sqsubseteq B\}$
for all $\{\langle D_1, D_1'\rangle, \langle D_2, D_2'\rangle\} \in \mathsf{ConflictSets}$ do
    if no $D$ such that $\mathcal{O}_2 \models_{\mathsf{rdfs}} D \sqsubseteq D_1' \sqcap D_2'$ exists then
        if $D_1' \sqsubseteq \neg D_2' \in \mathcal{O}_2$ and $C$ exist s.t. $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}' \models_{\mathsf{rdfs}} C \sqsubseteq D_1' \sqcap D_2'$ then
            $\mathsf{prune}(\mathcal{M}' \cup \mathcal{O}_2, \{\{\langle D_1, D_1'\rangle, \langle D_2, D_2'\rangle\}, \{D_1' \sqsubseteq \neg D_2'\}\})$
        else if $\mathsf{semSim}(D_1', D_2') \leq \mathsf{Config.Distance}.thr$ then
            $\mathsf{prune}(\mathcal{M}', \{\{\langle D_1, D_1'\rangle, \langle D_2, D_2'\rangle\}\})$
        end if
    end if
end for
return $\langle \mathcal{O}_2, \mathcal{M}'\rangle$
```

An effort to construct a large medical KB by integrating existing medical sources recently started in Babylon Health.[3] The KB would serve as the backbone for healthcare services (diagnosis, drug prescription, and more) as well as for other tasks like medical text annotation, understanding, and reasoning. For these purposes a modular and highly configurable ontology integration framework was implemented which is using ontology mapping to discover correspondences between a new medical source and the current KB and enrich the latter with new medical knowledge. There were two major requirements in this effort. First, integrating new sources should not affect the behaviour of the services already functioning with the KB, hence its structure should not change when new sources are integrated. Moreover, the KB should not contain many entities with a large label overlap as this complicates text annotation tasks as well as doctors who are selecting classes form the KB for diagnostic purposes. To address these requirements our framework is using the notion of conservativity for tracking the structural changes, however, in order to repair them we propose a novel fine-grained approach which avoids dropping mappings as much as possible.

First, violations stemming from mappings of higher-multiplicity (i.e., those that map two entities from one ontology to the same entity in the other) are separated from the rest and both are treated differently since they are of different nature. The former are repaired by altering the mappings, however, the latter are repaired by dropping *axioms* from the new ontology. Our motivation is that services have already committed to the structure of the KB and parts of the new ontology that are in disagreement with this conceptualisation can be dropped. In

[3] https://www.babylonhealth.com/.

addition, this approach helps reduce ambiguity and duplication of the integrated ontology as much as possible. Regarding violations on the structure of the new ontology, again a distinction between mappings of higher-multiplicity and the rest is made. The former are repaired by dropping mappings, however, the latter can be allowed since, as stated, the hierarchy of the new ontology is given low priority and various heuristics proposed in the literature can be used to guide this process. We formalised our framework using the notion of a (maximal) safe extension of a KB and provided an exact algorithm that is based on computing all repair plans [6].

Unfortunately, detecting all repair plans is known to be computationally very expensive [8,12]. Consequently, we next present a concrete implementation of our framework which is using approximate but efficient algorithms for violation detection (all state-of-the-art systems are based on approximate algorithms). Our implementation has already been used to create a medical KB by integrating SNOMED, NCI, FMA, and CHV, and is currently under use within Babylon. We conclude the paper with an experimental evaluation and a comparison against state-of-the-art mapping repair systems obtaining encouraging results. In more detail, our implementation is currently the only one that can apply a general conservativity-based mapping repair strategy (not only mapping coherency detection) on such a large KB while the created KB contains far less distinct classes with overlapping labels (i.e., less ambiguity and duplication). In addition, it was the only approach for which no conservativity violations could be detected in the integrated KB.

2 Ontologies and Ontology Matching

For brevity reasons, throughout the paper we will use Description Logic notation. For a set of real numbers S we use $\oplus S$ to denote the sum of its elements. For p an ontology prefix and C some class we sometimes write $p{:}C$ to denote that C appears in ontology with prefix p. Hence, for IRI prefixes $p_1 \neq p_2$, $p_1{:}C$ and $p_2{:}C$ denote different classes. For an ontology $\mathcal{O}$ we use $\mathsf{Sig}(\mathcal{O})$ to denote the set of classes that appear in $\mathcal{O}$. Given an ontology $\mathcal{O}$ we assume that all classes C in $\mathcal{O}$ have at least one triple of the form $\langle C\ \mathsf{skos{:}prefLabel}\ v\rangle$ and zero or more triples of the form $\langle C\ \mathsf{skos{:}altLabel}\ v_i\rangle$. For a given class C function $\mathsf{pref}(C)$ returns the string value v in the triple $\langle C\ \mathsf{skos{:}prefLabel}\ v\rangle$. An ontology is called *coherent* if every $C \in \mathsf{Sig}(\mathcal{O})$ with $C \neq \bot$ is satisfiable.

In the literature, the notion of a Knowledge Base is almost identical to that of an ontology, i.e., a set of axioms describing the entities of a domain. In the following, we loosely use the term "Knowledge Base" ($\mathcal{KB}$) to mean a possibly large ontology that has been created by integrating various other ontologies but, formally speaking, a $\mathcal{KB}$ is an OWL ontology.

Ontology matching (or *ontology alignment*) is the process of discovering correspondences (mappings) between the entities of two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. To represent mappings we use the formulation presented in [12]. That is, a mapping between $\mathcal{O}_1$ and $\mathcal{O}_2$ is a 4-tuple of the form $\langle C, D, \rho, n\rangle$, where $C \in \mathsf{Sig}(\mathcal{O}_1)$

$D \in \mathsf{Sig}(\mathcal{O}_2)$, $\rho \in \{\equiv, \sqsupseteq, \sqsubseteq\}$ is the *mapping type*, and $n \in (0, 1]$ is the confidence value of the mapping. Moreover, we interpret mappings as DL axioms—that is, $\langle C, D, \rho, n\rangle$ can be seen as the axiom $C\ \rho\ D$ with the degree attached as an annotation. Hence, for a mapping $\langle C, D, \rho, n\rangle$ when we write $\mathcal{O} \cup \{\langle C, D, \rho\rangle\}$ we mean $\mathcal{O} \cup \{C\ \rho\ D\}$ while for a set of mappings $\mathcal{M}$, $\mathcal{O} \cup \mathcal{M}$ denotes the set $\mathcal{O} \cup \{m \mid m \in \mathcal{M}\}$. When not relevant and for simplicity we will often omit ρ and n and simply write $\langle C, D\rangle$. A matcher is an algorithm that takes as input two ontologies and returns a set of mappings.

3 An Ontology Integration Framework

Large KBs can be constructed by integrating existing, complementary, and possibly overlapping ontologies. For example, in the biomedical domain, ontologies for diseases, drugs, drug side-effects, genes, and so on, exist that can be integrated in order to build a large medical KB. However, before putting two sources together it would be beneficial to discover their overlapping parts and establish mappings between their equivalent entities.

Example 1. Consider an ontology-based medical application that is using the SNOMED ontology $\mathcal{O}_{\mathsf{snmd}}$ as a KB. Although SNOMED is a large and well-engineered ontology it is still missing medical information like textual definitions for all classes as well as relations between diseases and symptoms. For example, for class the notion of "Ewing Sarcoma" SNOMED only contains the axiom $\mathsf{snmd{:}EwingSarcoma} \sqsubseteq \mathsf{snmd{:}Sarcoma}$ and no relations to signs or symptoms. In contrast, the NCI ontology $\mathcal{O}_{\mathsf{nci}}$ contains the following axiom about this disease:

$$\mathsf{nci{:}EwingSarcoma} \sqsubseteq \exists \mathsf{nci{:}mayHaveSymptom.nci{:}Fever}$$

We can use ontology matching to establish links between the related entities in $\mathcal{O}_{\mathsf{snmd}}$ and $\mathcal{O}_{\mathsf{nci}}$ and then integrate the two sources in order to enrich our KB. More precisely, using the labels of the aforementioned classes we can identify the following mappings:

$$m_1 = \langle \mathsf{snmd{:}EwingSarcoma}, \mathsf{nci{:}EwingSarcoma}, \equiv\rangle$$
$$m_2 = \langle \mathsf{snmd{:}Fever}, \mathsf{nci{:}Fever}, \equiv\rangle$$

and hence replace our KB with $\mathcal{O}'_{\mathsf{snmd}} := \mathcal{O}_{\mathsf{snmd}} \cup \mathcal{O}_{\mathsf{nci}} \cup \{m_1, m_2\}$. Then, $\mathcal{O}'_{\mathsf{snmd}}$ contains the knowledge that "Ewing sarcoma may have fever as a symptom". ♢

Unfortunately, it is well-known that integrating ontologies using the initially computed mappings can lead to unexpected consequences like, introducing unsatisfiable classes [11] or structural changes to the input ontologies [6].

Example 2. Consider again the SNOMED and NCI ontologies. Both ontologies contain classes for the notion of "soft tissue disorder" and "epicondylitis". Hence, it is reasonable for a matching algorithm to compute the following mappings:

$$m_1 = \langle \mathsf{snmd{:}SoftTissueDisorder}, \mathsf{nci{:}SoftTissueDisorder}, \equiv\rangle$$
$$m_2 = \langle \mathsf{snmd{:}Epicondylitis}, \mathsf{nci{:}Epicondylitis}, \equiv\rangle$$

Algorithm 3. $\mathsf{KnowledgeBaseConstruction}(\mathcal{KB}, \mathcal{O}, \mathsf{Config})$

```
Input: The current KB $\mathcal{KB}$, a new ontology $\mathcal{O}$ and a configuration $\mathsf{Config}$.
$\mathsf{Mappings} := \emptyset$
for all $\mathsf{matcher} : \mathsf{Config.Align.Matchers}$ do
    for all $\langle C, D, \rho, n \rangle \in \mathsf{matcher}(\mathcal{KB}, \mathcal{O})$ do
        $\mathsf{Mappings} := \mathsf{Mappings} \cup \{\langle C, D, \rho, n, \mathsf{matcher} \rangle\}$
    end for
end for
$\mathcal{M}_f := \emptyset$
$w = \oplus\{\mathsf{matcher}.w \mid \mathsf{matcher} \in \mathsf{Config.Align.Matchers}\}$
for all $\langle C, D, \rho, \_, \_ \rangle \in \mathsf{Mappings}$ such that no $\langle C, D, \rho, n \rangle$ exists in $\mathcal{M}_f$ do
    $n := \oplus\{n_i \times \mathsf{matcher}.w \mid \langle C, D, \rho, n_i, \mathsf{matcher} \rangle \in \mathcal{M}\}/w$
    if $n \geq \mathsf{Config.Align}.thr$ then
        $\mathcal{M}_f := \mathcal{M}_f \cup \{\langle C, D, \rho, n \rangle\}$
    end if
end for
$\langle \mathcal{O}', \mathcal{M}_f \rangle := \mathsf{postProcessNewOntoStructure}(\mathcal{KB}, \mathcal{O}, \mathcal{M}_f, \mathsf{Config})$
$\langle \mathcal{O}', \mathcal{M}_f \rangle := \mathsf{postProcessKBStructure}(\mathcal{KB}, \mathcal{O}', \mathcal{M}_f, \mathsf{Config})$
return $\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}_f$
```

However, in NCI we have $\mathcal{O}_{\mathsf{nci}} \models \mathsf{nci{:}Epicondylitis} \sqsubseteq \mathsf{nci{:}SoftTissueDisorder}$ while in SNOMED $\mathcal{O}_{\mathsf{snmd}} \not\models \mathsf{snmd{:}Epicondylitis} \sqsubseteq \mathsf{snmd{:}SoftTissueDisorder}$. Hence, in the integrated ontology we will have:

$$\mathcal{O}_{\mathsf{snmd}} \cup \mathcal{O}_{\mathsf{nci}} \cup \{m_1, m_2\} \models \mathsf{snmd{:}Epicondylitis} \sqsubseteq \mathsf{snmd{:}SoftTissueDisorder}$$

introducing a relation between classes of $\mathcal{O}_{\mathsf{snmd}}$ that did not originally hold and which can have a significant impact on the services of our application which are already based on the structure of $\mathcal{O}_{\mathsf{snmd}}$. ◇

The amount of such structural changes can be captured by the notion of logical difference [9]. Like in [6] for performance reasons we also use an approximate version of logical difference formalised next.

Definition 1 ([6]). *Let A, B be atomic classes (including $\top, \bot$), let Σ be a signature and let $\mathcal{O}$ and $\mathcal{O}'$ be two OWL 2 ontologies. The approximation of the Σ-deductive difference between $\mathcal{O}$ and $\mathcal{O}'$ (denoted $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}, \mathcal{O}')$) as the set of axioms of the form $A \sqsubseteq B$ satisfying: (i) $A, B \in \Sigma$, (ii) $\mathcal{O} \not\models A \sqsubseteq B$, and (iii) $\mathcal{O}' \models A \sqsubseteq B$.*

Using logical difference, the notion of a *conservative alignment* has been proposed in the literature [6,19,20] which dictates that for two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ and for $\Sigma_1 = \mathsf{Sig}(\mathcal{O}_1)$ and $\Sigma_2 = \mathsf{Sig}(\mathcal{O}_2)$ the set of mappings $\mathcal{M}$ must be such that $\mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M})$ and $\mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}_2, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M})$ are empty. An axiom belonging to either of these sets is called a *(conservativity) violation* and can be "repaired" by removing mappings form the initially computed sets.

Based on the above we have designed a Knowledge Base construction algorithm that is depicted in Algorithm 3. The algorithm accepts as input the current KB $\mathcal{KB}$, a new ontology $\mathcal{O}$ which will be used to enrich $\mathcal{KB}$ and a configuration Config. The configuration is used to tune and change various parameters like thresholds etc., many of which will be described in the rest of the paper. In brief, the algorithm first applies a set of matchers in order to compute a set of mappings between $\mathcal{KB}$ and $\mathcal{O}$ (lines 1–6). The set of matchers to be used is specified in the configuration object ($\mathsf{Config.Align.Matchers}$) and each of them has a different weight assigned ($\mathsf{matcher}.w$). After all matcher have finished, the mappings are aggregated and a threshold is applied ($\mathsf{Config.Align}.thr$) in order to keep only mappings with a high confidence (lines 7–14). As mentioned previously, these mappings are still not the final ones since they may cause conservativity violations. These are handled by two functions, namely $\mathsf{postProcessNewOntoStructure}$ and $\mathsf{postProcessKBStructure}$ which are discussed in detail in the next section.

4 Safe Ontology Integration

The typical approach to resolve conservativity violations so far has been to remove mappings [6,8,19,20]. However, this approach may introduce other issues like having distinct classes with a large overlap in their labels, hence introducing redundancy and ambiguity. Assume for instance, that in Example 2 we drop mapping m_2. Then, the integrated ontology will contain two different classes for the real-world notion of “epicondylitis” (i.e., $\mathsf{nci{:}Epicondylitis}$ and $\mathsf{snmd{:}Epicondylitis}$) each with overlapping labels. Subsequently, a service that is using the former class internally cannot interoperate with a service that is using the latter as there is no axiom specifying that the two classes are actually the same.

Instead of removing mappings, another way to repair a violation is by removing axioms from one of the input ontologies.

Example 3. Consider again Example 2 where $\mathcal{O}_{\mathsf{snmd}}$ serves as the current version of the application KB. Instead of computing $\mathcal{KB}_1^{\mathsf{int}} := \mathcal{O}_{\mathsf{snmd}} \cup \mathcal{O}_{\mathsf{nci}} \cup \{m_1, m_2\}$ as in Example 2 assume that we compute the following:

$$\mathcal{KB}_2^{\mathsf{int}} := \mathcal{KB}_1^{\mathsf{int}} \setminus \{\mathsf{nci{:}Epicondylitis} \sqsubseteq \mathsf{nci{:}SoftTissueDisorder}\}$$

Then, we have $\mathcal{KB}_2^{\mathsf{int}} \not\models \mathsf{snmd{:}Epicondylitis} \sqsubseteq \mathsf{snmd{:}SoftTissueDisorder}$ and hence $\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_{\mathsf{snmd}})}(\mathcal{O}_{\mathsf{snmd}}, \mathcal{KB}_2^{\mathsf{int}}) = \emptyset$ as desired. ♢

This approach is reasonable if we assume that an application is already using some Knowledge Base and the role of new ontologies is to enrich and extend it with new information but without altering its structure. Then, parts of the new ontology that cause violations can be dropped.

However, not all violations can be repaired by removing axioms from $\mathcal{O}_2$. This is the case for mappings of higher multiplicity, i.e., those that map two different classes of one ontology to the same class in the other.

Example 4. Consider again ontology $\mathcal{O}_{\mathsf{snmd}}$ and $\mathcal{O}_{\mathsf{nci}}$. SNOMED contains classes Eczema and AtopicDermatitis whereas NCI contains class Eczema that also has "Atopic Dermatitis" as an alternative label. Hence, a matching algorithm could create two mappings of the form:

$$\begin{aligned} m_1 &= \langle \mathsf{snmd{:}Eczema}, \mathsf{nci{:}Eczema}, \equiv\rangle \\ m_2 &= \langle \mathsf{snmd{:}AtopicDermatitis}, \mathsf{nci{:}Eczema}, \equiv\rangle \end{aligned}$$

which imply that snmd:Ezcema and snmd:AtopicDermatitis are equivalent although this is not the case in $\mathcal{O}_{\mathsf{snmd}}$. ♢

In these cases it is clear that the only way to repair such violations is by altering the mapping set. One approach would be to drop one of the two mappings or perhaps even change their type from $\equiv$ to $\sqsubseteq$ or $\sqsupseteq$ and we argue that the choice is case dependent. In the previous example, we may decide that SNOMED is more granular than NCI in the sense that Atopic Dermatitis is a type of Eczema whereas the NCI term captures a more general notion. Hence, we may decide to change the mappings to $\langle \mathsf{snmd{:}Eczema}, \mathsf{nci{:}Eczema}, \sqsubseteq\rangle$ and $\langle \mathsf{snmd{:}AtopicDermatitis}, \mathsf{nci{:}Eczema}, \sqsubseteq\rangle$. However, we may also conclude that the alternative labels in NCI don't strictly denote synonym (alternative) terms for diseases but rather similar ones and hence decide to drop mapping m_2.

Based on the above we introduce the notion of a safe extension of an ontology.

Definition 2. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two ontologies and let $\mathcal{M}$ be a set of mappings computed between them. The* safe extension *of $\mathcal{O}_1$ w.r.t. $\mathcal{O}_2, \mathcal{M}$ is a pair $\langle \mathcal{O}', \mathcal{M}'\rangle$ such that $\mathcal{O}' \subseteq \mathcal{O}_2, \mathcal{M}' \subseteq \mathcal{M}$ and $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}') = \emptyset$ for $\Sigma = \mathsf{Sig}(\mathcal{O}_1)$.*

The pair of an empty ontology and set of mappings $(\langle \emptyset, \emptyset\rangle)$ is a trivial safe extension but one is usually interested in some maximal safe extension similar to the notion of diagnosis in mapping repair [8].

Definition 3. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two ontologies and let $\mathcal{M}$ be a set of mappings computed between them. A safe extension $\langle \mathcal{O}', \mathcal{M}'\rangle$ of $\mathcal{O}_1$ w.r.t. $\mathcal{O}_2, \mathcal{M}$ is called* maximal *if no safe extension $\langle \mathcal{O}'', \mathcal{M}''\rangle$ exists s.t. either $\mathcal{O}'' \supset \mathcal{O}'$ or $\mathcal{M}'' \supset \mathcal{M}'$.*

Motivated by the above we have designed Algorithm 4 that accepts as input two ontologies $\mathcal{O}_1, \mathcal{O}_2$ and returns a subset of $\mathcal{O}_2$ and a subset of $\mathcal{M}$ in an attempt to compute a safe extension. The algorithm first processes mappings of higher multiplicity w.r.t. entities in $\mathcal{O}_1$ using function disambiguate-m-1 whose properties are formalised next.

Definition 4. *Given a set of mappings $\mathcal{M} = \{\langle C_1, D\rangle, \langle C_2, D\rangle, \ldots \langle C_n, D\rangle\}$ function* disambiguate-m-1 *returns a set $\mathcal{M}' \subseteq \mathcal{M}$ that satisfies the following property: it contains either a single mapping of the form $\langle C_i, D, \equiv\rangle$ or only mappings of the form $\langle C_i, D, \sqsupseteq\rangle$.*

Afterwards, the algorithm calls algorithm allPlans [6] passing the appropriate parameters in order to compute sets of axioms each of which has the following

Algorithm 4. $\mathsf{postProcessKBStructure}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}, \mathsf{Config})$

```
Input: Two coherent ontologies $\mathcal{O}_1, \mathcal{O}_2$, a set of mappings $\mathcal{M}$, and a Config object.
$\mathcal{M}_{\text{m-1}} := \{\langle C_i, D\rangle \mid \{\langle C_i, D\rangle, \langle C_j, D\rangle\} \subseteq \mathcal{M} \wedge C_i \neq C_j\}$.
$\mathcal{M}' := \mathcal{M} \setminus \mathcal{M}_{\text{m-1}}$
for all $D \in \mathsf{Sig}(\mathcal{O}_2)$ do
    $\mathcal{M}' := \mathcal{M}' \cup \mathsf{disambiguate\text{-}m\text{-}1}(\{\langle C_i, D\rangle \mid \langle C_i, D\rangle \in \mathcal{M}_{\text{m-1}}\}, \mathsf{Config})$
end for
$\mathsf{P} := \mathsf{allPlans}(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}', \mathcal{O}_2, \emptyset, \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}'))$ where $\Sigma = \mathsf{Sig}(\mathcal{O}_1)$
Pick some $P \in \mathsf{P}$ such that no $P' \in \mathsf{P}$ exists with $P' \subset P$
return $\langle \mathcal{O}_2 \setminus P, \mathcal{M}' \rangle$
```

property: if it is removed from $\mathcal{O}_2$ then all violations would be repaired. From all those sets the algorithm picks some minimal one (w.r.t. $\subseteq$) and removes it from $\mathcal{O}_2$.

Lemma 1. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two coherent ontologies and let $\mathcal{M}$ be a set of mappings between them such that every class in $\mathcal{O}_2$ is satisfiable in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$. When applied on $\mathcal{O}_1, \mathcal{O}_2$ and $\mathcal{M}$ Algorithm 4 returns a maximal safe extension of $\mathcal{O}_1$ w.r.t. $\mathcal{O}_2, \mathcal{M}$.*

Although, we are strict with respect to violations that are implied by the mappings to the structure of the KB, we can be more relaxed with respect to violations over the ontology that is being used for the enrichment. Several heuristics have been presented in the literature in order to decide which violations to allow and which to repair. A violation $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_2)}(\mathcal{O}_2, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M})$ may be allowed if A and B are somehow semantically related, e.g., if A and B have a common descendant [19]. In contrast, a violation should be repaired if $\mathcal{O}_2 \models A \sqsubseteq \neg B$, i.e., A and B are disjoint [11] or if the assumption of disjointness [13] can be applied to them—that is, if A and B are in different (distant) parts of the hierarchy of $\mathcal{O}_2$ and hence we can assume that they are disjoint.

Motivated by the above we have designed Algorithm 5. Like before mappings of higher multiplicity are treated separately by function $\mathsf{disambiguate\text{-}1\text{-}m}$. Afterwards, the algorithm iterates over all violations w.r.t. ontology $\mathcal{O}_2$ and uses many of the aforementioned heuristics, like common descendants (line 8), unsatisfiability of classes (lines 9–12) and semantic or taxonomical similarity using function semSim and a pre-defined threshold $\mathsf{Config.Distance}.thr$ (lines 13–16) in order to decide to repair them or not. To figure out how to repair a violation algorithm $\mathsf{allPlans}$ is utilised again. This time $\mathcal{M}'$ (and possibly $\mathcal{O}_2$) instead of $\mathcal{O}_1$ is passed as a second parameter since we don't want the plans to contain axioms from $\mathcal{O}_1$. Note that in case some class A is unsatisfiable in the integrated ontology either mappings from $\mathcal{M}'$ or axioms from $\mathcal{O}_2$ that lead to this unsatisfiability may be selected to be removed. This choice was motivated by the conceptual differences between NCI and FMA regarding anatomical structures and proteins which are disjoint in one ontology but semantically related in the other. The choice of which plan to pick to remove is again case dependent hence we abstract this away using the function prune which picks at-least one plan.

Algorithm 5. $\mathsf{postProcessNewOntoStructure}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}, \mathsf{Config})$

```
Input: Ontologies $\mathcal{O}_1, \mathcal{O}_2$, set of mappings $\mathcal{M}$, and $\mathsf{Config}$ object.
$\mathcal{M}_{\text{1-m}} := \{\langle C, D_i \rangle \mid \{\langle C, D_i \rangle, \langle C, D_j \rangle\} \subseteq \mathcal{M} \wedge D_i \neq D_j\}$.
$\mathcal{M}' := \mathcal{M} \setminus \mathcal{M}_{\text{1-m}}$
for all $C \in \mathsf{Sig}(\mathcal{O}_1)$ do
    $\mathcal{M}' := \mathcal{M}' \cup \mathsf{disambiguate\text{-}1\text{-}m}(\{\langle C, D_i \rangle \mid \langle C, D_i \rangle \in \mathcal{M}_{\text{1-m}}\}, \mathsf{Config})$
end for
$\mathsf{LDiff} := \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_2, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}')$ where $\Sigma = \mathsf{Sig}(\mathcal{O}_2)$
for all $A \sqsubseteq B \in \mathsf{LDiff}$ do
    if no $C$ such that $\mathcal{O}_2 \models C \sqsubseteq A \sqcap B$ exists then
        if $B = \bot$ then
            $\mathcal{O}_2^{\neg} := \{C \sqsubseteq \neg D \mid C \sqsubseteq \neg D \in \mathcal{O}_2\}$
            $\mathsf{P} := \mathsf{allPlans}(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}', \mathcal{O}_2^{\neg} \cup \mathcal{M}', \emptyset, \{A \sqsubseteq \bot\})$
            $\mathsf{prune}(\mathcal{M}' \cup \mathcal{O}_2, \mathsf{P})$
        else if $\mathsf{semSim}(A, B) \leq \mathsf{Config.Distance}.thr$ then
            $\mathsf{P} := \mathsf{allPlans}(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}', \mathcal{M}', \emptyset, \{A \sqsubseteq B\})$
            $\mathsf{prune}(\mathcal{M}', \mathsf{P})$
        end if
    end if
end for
return $\langle \mathcal{O}_2, \mathcal{M}' \rangle$
```

Lemma 2. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two coherent ontologies and let $\mathcal{M}$ be a set of mappings between them. When applied on $\mathcal{O}_1, \mathcal{O}_2$ and $\mathcal{M}$ Algorithm 5 returns a pair $\langle \mathcal{O}', \mathcal{M}' \rangle$ such that every class in $\mathcal{O}'$ is satisfiable in $\mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}'$.*

Using Lemmas 1 and 2 we can show the main result of our paper.

Theorem 1. *Let $\mathcal{KB}$ and $\mathcal{O}$ be two coherent ontologies, let* Config *be some configuration and let $\mathcal{KB}'$ be the output of Algorithm 3 when applied on $\mathcal{KB}, \mathcal{O}$ and* Config. *Then the following hold:*

1. $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{KB}, \mathcal{KB}') = \emptyset$ *where* $\Sigma = \mathsf{Sig}(\mathcal{KB})$.
2. $\mathcal{KB}'$ *is coherent.*

5 Practical Algorithms

We have provided with concrete implementations of the algorithms and functions presented in the previous section. Regarding matching (lines 2–6 of Algorithm 3), two in-house label-based matchers have been implemented, namely ExactLabelMatcher and FuzzyStringMatcher. The former builds an inverted index of class labels [3] after some string normalisations, like removing possessive cases (e.g., Alzheimer's) and singularisation [10], and matches ontologies using these indexes. The latter is based on the ISub string similarity metric [21]. Since this algorithm does not scale well on large inputs [3] it is mostly used for disambiguating higher-multiplicity mappings or if we wish to re-score subsets of mappings

Algorithm 6. $\mathsf{planApproximation}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$

```
Input: Two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ a set of mappings between them.
Exclusions := $\emptyset$
ConflictSets := $\{\{m_1, m_2\} \mid \mathcal{O}_1 \cup \mathcal{O}_2 \cup \{m_1, m_2\} \models_{\mathsf{rdfs}} A \sqsubseteq B, \mathcal{O}_1 \not\models_{\mathsf{rdfs}} A \sqsubseteq B\}$
for all $\{\langle A, A'\rangle, \langle B, B'\rangle\} \in$ ConflictSets with $\mathcal{O}_2 \models_{\mathsf{rdfs}} A' \sqsubseteq B'$ do
    Exclusions := Exclusions $\cup \{A' \sqsubseteq E \mid A' \sqsubseteq E \in \mathcal{O}_2, \mathcal{O}_2 \models_{\mathsf{rdfs}} E \sqsubseteq B'\}$
end for
return Exclusions
```

with low confidence degrees. In addition to these matchers, the state-of-the-art systems AML [4] and LogMap [7] can also be used in Algorithm 3.

Regarding functions disambiguate-m-1 and disambiguate-1-m the following strategy has been implemented so far:

> For a set of mappings $\{\langle C_1, D\rangle, \langle C_2, D\rangle, \ldots, \langle C_n, D\rangle\}$ and some real-value threshold Config.Disamb.th, if $i \in [1, n]$ exists such that the following two conditions hold:
>
> 1. $\mathsf{ISub}(\mathsf{pref}(C_i), \mathsf{pref}(D)) > \mathsf{ISub}(\mathsf{pref}(C_j), \mathsf{pref}(D))$ for every $j \neq i$ and
> 2. $\mathsf{ISub}(\mathsf{pref}(C_i), \mathsf{pref}(D)) \geq$ Config.Disamb.th
>
> then return $\langle C_i, D\rangle$. Similarly in function disambiguate-1-m for sets of mappings of the form $\{\langle C, D_1\rangle, \langle C, D_2\rangle, \ldots, \langle C, D_n\rangle\}$.

A major practical consideration of Algorithms 4 and 5 is the call to algorithm allPlans. This algorithm does not scale in practice since it iterates over the power-set of the second parameter (i.e., $\mathcal{O}_2$) [6]. Consequently, as usually done in literature [8,19] we are using approximations of plan computation and violation repair. More precisely, lines 6 and 7 in Algorithm 4 are replaced by the call $P :=$ $\mathsf{planApproximation}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}')$ where the implementation of this function is given in Algorithm 6 and is inspired by the Alcomo repair algorithm [8,12]. This algorithm is based on the assumption that logical differences of the form $A \sqsubseteq B$ stem from *exactly* two mappings which map classes A and B for which $\mathcal{O}_1 \not\models A \sqsubseteq B$ to classes A' and B' in $\mathcal{O}_2$ for which a path of SubClassOf axioms in $\mathcal{O}_2$ exists ($\models_{\mathsf{rdfs}}$), hence implying changes in the structure of $\mathcal{O}_1$. Although in theory this may not always be the case, most violations in practice do follow this pattern. For every such pair of mappings the algorithm picks to remove from $\mathcal{O}_2$ some axiom of the form $A' \sqsubseteq E$, i.e., it tries in some sense to remove the "weakest" axiom from $\mathcal{O}_2$. This choice is motivated by belief revision and the *principle of minimal change* [1].

Example 5. Consider for example the following two ontologies:

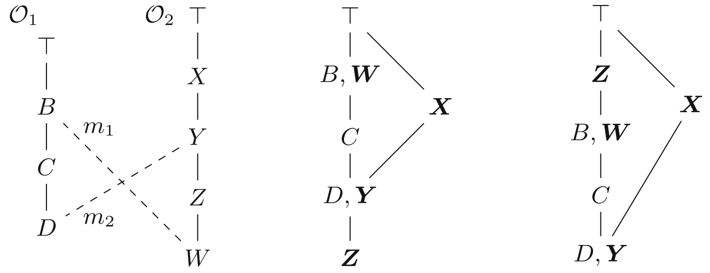
$$\mathcal{O}_1 = \{D \sqsubseteq C, C \sqsubseteq B\}$$
$$\mathcal{O}_2 = \{W \sqsubseteq Z, Z \sqsubseteq Y, Y \sqsubseteq X\}$$

and assume the set of mappings $\mathcal{M} = \{m_1, m_2\}$ where $m_1 = \langle D, Y\rangle$ and $m_2 = \langle B, W\rangle$. Clearly, for $\Sigma = \mathsf{Sig}(\mathcal{O}_1)$ we have $B \sqsubseteq D \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M})$ and we can repair this violation by either removing $ax_1 = W \sqsubseteq Z$ or $ax_2 = Z \sqsubseteq Y$.

Algorithm 7. $\mathsf{newOntologyApproximate}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$

```
$\mathsf{ConflictSets} := \{\{m_1, m_2\} \mid \mathcal{O}_1 \cup \mathcal{O}_2 \cup \{m_1, m_2\} \models_{\mathsf{rdfs}} A \sqsubseteq B, \mathcal{O}_2 \not\models A \sqsubseteq B\}$
for all $\{\langle D_1, D_1' \rangle, \langle D_2, D_2' \rangle\} \in \mathsf{ConflictSets}$ do
    if no $D$ such that $\mathcal{O}_2 \models_{\mathsf{rdfs}} D \sqsubseteq D_1' \sqcap D_2'$ exists then
        if $D_1' \sqsubseteq \neg D_2' \in \mathcal{O}_2$ and $C$ exist s.t. $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}' \models_{\mathsf{rdfs}} C \sqsubseteq D_1' \sqcap D_2'$ then
            $\mathsf{prune}(\mathcal{M}' \cup \mathcal{O}_2, \{\{\langle D_1, D_1' \rangle, \langle D_2, D_2' \rangle\}, \{D_1' \sqsubseteq \neg D_2'\}\})$
        else if $\mathsf{semSim}(D_1', D_2') \leq \mathsf{Config.Distance}.thr$ then
            $\mathsf{prune}(\mathcal{M}', \{\{\langle D_1, D_1' \rangle, \langle D_2, D_2' \rangle\}\})$
        end if
    end if
end for
```

Ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ as well as KBs $\mathcal{KB}^{\mathsf{int}}_{ax_1} = \mathcal{O}_1 \cup \mathcal{O}_2 \setminus \{ax_1\} \cup \mathcal{M}$ and $\mathcal{KB}^{\mathsf{int}}_{ax_2} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \setminus\{ax_1\} \cup \mathcal{M}$ are depicted graphically in Fig. 1, where solid lines denote subclass relations, and dashed lines the two mappings. As we can see, although both integrated ontologies do not exhibit violations over $\mathcal{O}_1$, the two cases differ in the amount of changes they impose on the classes of $\mathcal{O}_1$. More precisely, for $\mathbf{S}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}) = \{A \sqsubseteq B \mid A \in \mathsf{Sig}(\mathcal{O}_1), \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M} \models A \sqsubseteq B\}$ we have $\mathbf{S}(\mathcal{O}_1, \mathcal{O}_2 \setminus \{ax_2\}, \mathcal{M}) \setminus \mathbf{S}(\mathcal{O}_1, \mathcal{O}_2 \setminus \{ax_1\}, \mathcal{M}) = \{B \sqsubseteq Z, C \sqsubseteq Z, D \sqsubseteq Z, D \sqsubseteq X\}$. Indeed, in this scenario Algorithm 5 will compute $\mathsf{Exclusions} := \{ax_1\}$. ♢



(a) Input Ontologies (b) $\mathcal{KB}^{\mathsf{int}}_1$ (c) $\mathcal{KB}^{\mathsf{int}}_2$

Fig. 1. Ontologies of Example 5 and resulting KBs; with bold we denote classes from $\mathcal{O}_2$.

Following a similar approach, the block of lines 6–18 in Algorithm 5 is replaced by the steps depicted in Algorithm 7. Again these steps assume that violations stem from pairs of "conflicting" mappings like those mentioned above. Similarly to Algorithm 5, we are again using the heuristics of common descendants, disjoint classes and class similarity as a guide for repairing the violations; all entailments are checked using RDFS-entailment.

Using Algorithm 3 and the techniques presented above we have started building a large medical KB to be used within Babylon Health. We used the SNOMED

January 2018 release (which contains 340K classes and 511K SubClassOf axioms) as a starting seed KB ($\mathcal{KB}_1$) and have so far iteratively integrated the following ontologies: NCI version 17.12d (which contains 130K classes and 143K SubClassOf axioms), CHV latest version from 2011 (which contains 57 K classes and 0 SubClassOf axioms) and FMA version 4.6.0 (which contains 104K classes and 255K SubClassOf axioms); all ontologies are from the official websites. As a matching algorithm we have so far used our ExactLabelMatcher. Statistics about the KBs that we created after each integration are depicted in Table 1. CHV is a flat list of layman terms of medical concepts. From that ontology we only integrated label information for the classes in CHV that mapped to some class in the Babylon KB; hence only data type properties increased in the KB in that step. The final KB is currently under use by various services within Babylon and we are in the process of also integrating the following resources: SNOMED drug extensions, ICD-10, Read Codes, MeSH, and more.

Table 1. Statistics about the KB after each integration/enrichment iteration.

	SNOMED	+NCI	+CHV	+FMA
Classes	340 995	429 241	429 241	524 837
Properties	93	124	124	219
SubClassOf axioms	511 656	617 542	617 542	713 313
ObjPropAssertions	526 146	664 742	664 742	962 190
DataPropAssertions	543 416	946 801	1 043 874	1 211 459

6 Evaluation

We have conducted an experimental evaluation in order to assess the effectiveness of our approach for integrating ontologies and remedying conservativity violations. Using SNOMED as our initial Knowledge Base we once integrated NCI and then FMA (starting again from scratch). We used our ExactLabelMatcher once with and once without the last post-processing steps in Algorithm 3 (lines 15 and 16). In the following we call the former setting bOWLing and the latter bOWLing$_n$. We used the latter setting as a "naive" baseline approach.

In addition, we also run Algorithm 3 using AML and two versions of LogMap called LogMap$_o$ and LogMap$_c$ in the following. AML and LogMap$_o$ repair mappings with respect to coherency, i.e., they only check for conservativity violations that lead to unsatisfiable classes. NCI contains 196 while FMA 33.5K disjoint classes axioms so this mapping repair is relevant. In contrast, LogMap$_c$ also checks for more general conservativity violations using the techniques presented in [19]. For all these systems we disabled the post-processing steps of Algorithm 3 in order to assess their mapping repair functionality. On the mapping sets computed by bOWLing$_n$ and LogMap$_o$ we have also run Alcomo [12] as a post-processing step. Alcomo is not a general matcher but a mapping repair system that can be used as a post-processing step. In the following we denote these settings as bOWLing$_n^{\mathsf{Alc}}$ and LogMap$_o^{\mathsf{Alc}}$.

Algorithm 3 did not terminate with AML and LogMap$_c$ after running for more than 16 h. As a second attempt we fragmented the ontologies into modules (using a $\sqsubseteq$-reachability based algorithm starting from top-level classes) and fed these one by one to Algorithm 3. For NCI we extracted 53 while for FMA 6 modules. Even in this case AML did not terminate when integrating FMA.

Table 2. Evaluation results

	SNOMED+NCI					
	$\vert\mathcal{M}\vert$	$\vert\mathcal{KB}^{\mathsf{int}}\vert$	$\vert\mathsf{LDiff}\vert$	Time	Loops	ambiguity
bOWLing$_n$	30 675	677 939	t.o.	12.7	127	16 708
bOWLing$_n^{\mathsf{Alc}}$	26 825	666 834	0.9m	35.9	100	17 177
bOWLing	19 258	638 702	0	12.2	0	7 810
LogMap$_o$	27 967	664 837	1.7m	120.9	74	17 632
LogMap$_o^{\mathsf{Alc}}$	27 763	664 354	1.5m	141.7	71	16 986
LogMap$_c$	21 838	433 711	897	54.4	0	8,266
AML	32 623	635 876	t.o.	75.0	298	14 353
	SNOMED+FMA					
	$\vert\mathcal{M}\vert$	$\vert\mathcal{KB}^{\mathsf{int}}\vert$	$\vert\mathsf{LDiff}\vert$	Time	Loops	ambiguity
bOWLing$_n$	8 809	614 728	240k	7.0	3	1 946
bOWLing$_n^{\mathsf{Alc}}$	7 886	615 291	93k	76.2	1	2 000
bOWLing	8 176	608 060	0	27.9	0	1 440
LogMap$_o$	7 334	615 252	117k	360.4	1	2 264
LogMap$_o^{\mathsf{Alc}}$	6 986	615 689	57k	428.4	1	2 253
LogMap$_c$	6 036	420 424	517	14 004.8	0	1 553

Our results are summarised in Table 2 where we give the number of computed mappings ($|\mathcal{M}|$), the number of SubClassOf axioms in the integrated ontology ($|\mathcal{KB}^{\mathsf{int}}|$), the number of axioms in $\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{KB})}(\mathcal{KB}, \mathcal{KB}^{\mathsf{int}})$ (denoted by $|\mathsf{LDiff}|$ and with "m" denoting millions), and the time to compute $\mathcal{KB}^{\mathsf{int}}$ (in minutes). Due to the very large size of the KB LDiff cannot be computed by any OWL reasoner so we computed the RDFS-level differences by simply traversing the SubClassOf hierarchy of the KB. In addition, we have also computed the following:

- number of cycles of the form $\{A_1 \sqsubseteq A_2, \ldots, A_n \sqsubseteq A_1\} \subseteq \mathcal{KB}^{\mathsf{int}}$. From a semantic point of view such cycles are not problematic, however, they do complicate graph-based algorithms like hierarchy traversal, extracting paths and depth counting, hence it is a design decision in Babylon to avoid them; input ontologies contain no cycles.

- a notion of "ambiguity" which we defined as the number of times a label appears in two different classes of a given ontology. We also calculated this metric over the original SNOMED, NCI, and FMA ontologies in order to measure their level of ambiguity. We obtained 1 055, 4 873, and 282, respectively, e.g., in SNOMED 1 055 labels appear in multiple classes.

First thing to note from the table is that all systems compute mapping sets of comparable size with the exception of bOWLing on SNOMED+NCI which computes a smaller mapping sets. This is mostly due to functions disambiguate-m-1 and disambiguate-1-m which prune mappings of higher-multiplicity. However, we should note that all mappings computed by this approach are one-to-one mappings, while in all other approaches from the roughly 27k mappings about 17k are actually one-to-one (i.e., fewer than those of bOWLing). The application of Alcomo on the mapping sets does remove some mappings in an attempt to repair the sets while LogMap_c that uses a general conservativity-based repairing approach also computes fewer mappings than LogMap_o.

As expected, the ontology produced by bOWLing contains fewer axioms due to the axiom exclusion strategy implemented in line 16 of Algorithm 3 which drops about 30% of NCI axioms and 10% of FMA axioms. However, the gains from this approach are apparent when considering other computed metrics. More precisely, the integrated ontology produced by bOWLing contains no axioms in LDiff in contrast to even more than 1 million new ancestor classes in some of the other approaches. Moreover, there are no cycles and, finally, a very low degree of ambiguity taking also into account the initial ambiguity of these ontologies (see above). The use of Alcomo as a post-processing step on $\mathsf{bOWLing}_n$ and LogMap_o does improve the numbers on these metrics, however, as it only focuses on coherency and not general conservativity it does not eliminate them completely. The only comparable approach is LogMap_c which computes a KB without cycles. However, LDiff is still not empty and the approach of dropping mappings increases the ambiguity metric. Recall that we were only able to run LogMap_c on the modules. Had it run on the whole ontology we believe the reported numbers would be higher since as one can note the integrated ontology in this module approach is also much smaller (almost 1/3 smaller). Finally, compared to all other systems our approach is much more scalable requiring a few minutes whereas in all other settings Algorithm 3 could take from one even up to 4 h (even when restricted to the modules). Note that in some cases we could not compute LDiff even after 12 h (t.o.).

7 Related Work and Conclusions

Constructing large Knowledge Bases (also called Knowledge Graphs these days) is a topic of intensive research and engineering the last years. The works [2,17] focus on extracting medical facts from text and use ontologies like UMLS and SNOMED mostly as flat vocabularies for performing named entity disambiguation, text annotation and information extraction. Hence, the focus is not on merging the medical knowledge and "fusing" the hierarchies. Malacards [14] is an effort for constructing a large disease KB by integrating information from many existing disease ontologies. To identify the overlaps between different sources a label-based unification algorithm is used which is very similar to our ExactLabelMatcher (labels are normalised, stemmed, singularised, etc. and a hash is created). However, this approach completely discards the hierarchy and the axioms of the original ontologies and the final output is a flat non-ontological structure.

In the current paper we have studied the problem of building large KBs from existing ontologies by integrating them and retaining as much of their initial structure and axioms as possible. Starting with an initial ontology as a seed KB we use new ontologies to extend and enrich it in an iterative way. Overlaps are discovered using ontology matching algorithms and mappings are post-processed in order to preserve properties of the structures of the KB and the new ontology. The algorithm is highly modular as different strategies for handling higher-multiplicity mappings can be implemented and different (or multiple) matchers can be used. Our post-processing steps are based on the notion of conservativity but differently than what is usually done in the literature [6,13,19] we propose to remove axioms from the new ontology in order to repair many of the violations. This is important in order to keep ambiguity low and not have many classes with overlapping labels. We have formalised our framework, designed an exact general algorithm and also presented concrete approximate and practical algorithms. These have already been used in Babylon Health to build a medical Knowledge Base (using SNOMED, NCI, FMA, and CHV) that forms the data and knowledge backbone of various clinical services. Finally, we have conducted an experimental evaluation comparing our conservativity repairing approach to state-of-the-art mapping repair systems obtaining very encouraging results. In summary, our results verify that ambiguity is very-low (almost none introduced compared to the initial ambiguity of the input ontologies), there were no detectable violations (LDiff), no cycles, and our algorithm scales.

8 Proofs

Lemma 1. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two coherent ontologies and let $\mathcal{M}$ be a set of mappings between them such that every class in $\mathcal{O}_2$ is satisfiable in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$. When applied on $\mathcal{O}_1, \mathcal{O}_2$ and $\mathcal{M}$ Algorithm 4 returns a minimal safe extension of $\mathcal{O}_1$ w.r.t. $\mathcal{O}_2, \mathcal{M}$.*

Proof. Let $\langle \mathcal{O}', \mathcal{M}' \rangle$ be the output of Algorithm 4 when applied on $\mathcal{O}_1, \mathcal{O}_2$, and $\mathcal{M}$. $\mathcal{O}' = \mathcal{O}_2 \setminus P$ for some $P \in \mathsf{P}$ and since the second parameter of the call to function $\mathsf{allPlans}$ is $\mathcal{O}_2$, then for every such P we have $P \subseteq \mathcal{O}_2$; hence $\mathcal{O}' \subseteq \mathcal{O}_2$.

Next we show that for $\Sigma = \mathsf{Sig}(\mathcal{O}_1)$ we have $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}') = \emptyset$. Let some $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M})$ and assume to the contrary that we have $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}')$. By definition we have $\mathcal{O}_1 \not\models A \sqsubseteq B$, hence $A \neq B$.

Consider some arbitrary justification $\mathcal{J}$ for $\mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}' \models A \sqsubseteq B$. If $\mathcal{J} \subseteq \mathcal{M}'$ then we have that $\mathcal{O}_1 \cup \mathcal{M}' \models A \sqsubseteq B$. Since $\mathcal{O}_1 \not\models A \sqsubseteq B$ and all mappings relate classes from $\mathcal{O}_1$ to classes in $\mathcal{O}_2$ this can only be the case if these mappings in $\mathcal{J}$ map two classes from $\mathcal{O}_1$ to the same class in $\mathcal{O}_2$: we can use resolution to prove $\mathcal{O}_1 \cup \mathcal{M}' \models A \sqsubseteq B$ in the process of which one of the mappings will introduce a symbol from $\mathcal{O}_2$; since every symbol of $\mathcal{O}_2$ is satisfiable in $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ it can only be removed by resolving it with another mapping that refers it and thus there must be two mappings that map two different classes of $\mathcal{O}_1$ to the same symbol in $\mathcal{O}_2$. However, this is not possible due to the properties of function $\mathsf{disambiguate\text{-}m\text{-}1}$ which has eliminated from $\mathcal{M}'$ mappings of higher multiplicity.

Hence, every justifications $\mathcal{J}$ of every violation $\alpha \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}')$, $\mathcal{J}$ must contain axioms from $\mathcal{O}_2$. But then, by Proposition 10 in [6] we have that P contains all P such that $\mathcal{O}_1 \cup \mathcal{O}_2 \setminus P \cup \mathcal{M} \not\models \alpha$, hence $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}') = \emptyset$.

Finally, the maximality condition on $\mathcal{O}'$ is satisfied by selecting some minimal w.r.t. $\subseteq$ plan from P in line 7. □

Lemma 2. *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two coherent ontologies and let $\mathcal{M}$ be a set of mappings between them. When applied on $\mathcal{O}_1, \mathcal{O}_2$ and $\mathcal{M}$ Algorithm 5 returns a pair $\langle \mathcal{O}', \mathcal{M}' \rangle$ such that every class in $\mathcal{O}'$ is satisfiable in $\mathcal{O}_1 \cup \mathcal{O}' \cup \mathcal{M}'$.*

Proof. Assume that some arbitrary class A is unsatisfiable in $\mathcal{O}' \cup \mathcal{M}'$. Since $\mathcal{O}_2$ is coherent and $\mathcal{O}' \subseteq \mathcal{O}_2$, then A is also satisfiable in $\mathcal{O}'$. Hence we must have that $A \sqsubseteq \bot \in \mathsf{LDiff}$. Again by coherency of $\mathcal{O}_2$ no C exists such that $\mathcal{O}_2 \models C \sqsubseteq A \sqcap \bot$, hence the algorithm enters block 9–12 and computes a plan for repairing this unsatisfiability ($A \sqsubseteq \bot$). Like in the proof of Lemma 1 a repair plan containing only mappings from $\mathcal{M}'$ exists. Due to the parameters used to call $\mathsf{allPlans}$ a repair plan containing disjointness axioms from $\mathcal{O}_2$ may also be present in P. Thus, the call to $\mathsf{allPlans}$ with second argument $\mathcal{M}' \cup \mathcal{O}_2$ returns some non-empty repair plan which is removed form $\mathcal{M}' \cup \mathcal{O}_2$ at line 12 repairing this unsatisfiability and contradicting the initial assumption. □

Theorem 1. *Let $\mathcal{KB}$ and $\mathcal{O}$ be two coherent ontologies, let* Config *be some configuration and let $\mathcal{KB}'$ be the output of Algorithm 3 when applied on $\mathcal{KB}, \mathcal{O}$ and* Config. *Then the following hold:*

1. $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{KB}, \mathcal{KB}') = \emptyset$ *where* $\Sigma = \mathsf{Sig}(\mathcal{KB})$.
2. $\mathcal{KB}'$ *is coherent.*

Proof. By Lemma 2 every class in $\mathcal{O}'$ that is passed as a second parameter in function $\mathsf{postProcessKBStructure}$ is satisfiable in $\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}'$. Hence, by Lemma 1 it follows that $\langle \mathcal{O}', \mathcal{M}' \rangle$ returned by Algorithm 4 in line 16 is a safe extension of $\mathcal{KB}$ hence item 1. holds.

For item 2, by Lemma 2 and since function $\mathsf{postProcessKBStructure}$ only removes axioms from $\mathcal{O}'$ it follows trivially that all classes of $\mathcal{O}'$ are satisfiable in $\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}'$. Consider some class A in $\mathcal{KB}$. Since $\mathcal{KB}$ is coherent then A is satisfiable in $\mathcal{KB}$ and by item 1. this class is also satisfiable in $\mathcal{O}'_{KG}$ for otherwise we would have $A \sqsubseteq \bot \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{KB}, \mathcal{KB}')$. □

References

1. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the logic of theory change: partial meet contraction and revision functions. J. Symb. Log. **50**(2), 510–530 (1985)
2. Ernst, P., Siu, A., Weikum, G.: KnowLife: a versatile approach for constructing a large knowledge graph for biomedical sciences. BMC Bioinform. **16**, 157:1–157:13 (2015)
3. Faria, D., Pesquita, C., Mott, I., Martins, C., Couto, F.M., Cruz, I.F.: Tackling the challenges of matching biomedical ontologies. J. Biomed. Semant. **9**(1), 4:1–4:19 (2018)
4. Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F., Couto, F.M.: The AgreementMakerLight ontology matching system. In: Meersman, R., Panetto, H., Dillon, T., Eder, J., Bellahsene, Z., Ritter, N., De Leenheer, P., Dou, D. (eds.) OTM 2013. LNCS, vol. 8185, pp. 527–541. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41030-7_38
5. Golbeck, J., Fragoso, G., Hartel, F.W., Hendler, J.A., Oberthaler, J., Parsia, B.: The national cancer institute's thésaurus and ontology. J. Web Semant. **1**(1), 75–80 (2003)
6. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Ontology integration using mappings: towards getting the right logical consequences. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 173–187. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02121-3_16
7. Jiménez-Ruiz, E., Grau, B.C., Zhou, Y.: LogMap 2.0: towards logic-based, scalable and interactive ontology matching. In: Proceedings of the 4th International Workshop on Semantic Web Applications and Tools for the Life Sciences, pp. 45–46 (2011)
8. Jiménez-Ruiz, E., Meilicke, C., Grau, B.C., Horrocks, I.: Evaluating mapping repair systems with large biomedical ontologies. In: Proceedings of the 26th International Workshop on Description Logics, pp. 246–257 (2013)
9. Konev, B., Walther, D., Wolter, F.: The logical difference problem for description logic terminologies. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) IJCAR 2008. LNCS (LNAI), vol. 5195, pp. 259–274. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-71070-7_21
10. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D.: The stanford CoreNLP natural language processing toolkit. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL, pp. 55–60 (2014)
11. Meilicke, C., Stuckenschmidt, H.: Applying logical constraints to ontology matching. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS (LNAI), vol. 4667, pp. 99–113. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74565-5_10
12. Meilicke, C., Stuckenschmidt, H.: An efficient method for computing alignment diagnoses. In: Polleres, A., Swift, T. (eds.) RR 2009. LNCS, vol. 5837, pp. 182–196. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-05082-4_13
13. Meilicke, C., Völker, J., Stuckenschmidt, H.: Learning disjointness for debugging mappings between lightweight ontologies. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 93–108. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87696-0_11

14. Rappaport, N., et al.: MalaCards: an integrated compendium for diseases and their annotation. Database (2013)
15. Salvadores, M., Alexander, P.R., Musen, M.A., Noy, N.F.: Bioportal as a dataset of linked biomedical ontologies and terminologies in RDF. Semant. Web **4**(3), 277–284 (2013)
16. Schriml, L.M.: Disease ontology: a backbone for disease semantic integration. Nucl. Acids Res. **40**(Database–Issue), 940–946 (2012)
17. Shi, L., Li, S., Yang, X., Qi, J., Pan, G., Zhou, B.: Semantic health knowledge graph: semantic integration of heterogeneous medical knowledge and services. BioMed Res. Int. 2017 (2017). (Article ID 2858423)
18. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Trans. Knowl. Data Eng. **25**(1), 158–176 (2013)
19. Solimando, A., Jiménez-Ruiz, E., Guerrini, G.: Detecting and correcting conservativity principle violations in ontology-to-ontology mappings. In: Mika, P., et al. (eds.) ISWC 2014. LNCS, vol. 8797, pp. 1–16. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11915-1_1
20. Solimando, A., Jiménez-Ruiz, E., Guerrini, G.: A multi-strategy approach for detecting and correcting conservativity principle violations in ontology alignments. In: 11th International Workshop on OWL: Experiences and Directions (2014)
21. Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 624–637. Springer, Heidelberg (2005). https://doi.org/10.1007/11574620_45
22. Vandenbussche, P., Atemezing, G., Poveda-Villalón, M., Vatant, B.: Linked open vocabularies (LOV): a gateway to reusable semantic vocabularies on the web. Semant. Web **8**(3), 437–452 (2017)