# A Multi-strategy Approach for Detecting and Correcting Conservativity Principle Violations in Ontology Alignments*

Alessandro Solimando[1], Ernesto Jiménez-Ruiz[2], and Giovanna Guerrini[1]

[1] DIBRIS, Università di Genova, Italy
[2] Department of Computer Science, University of Oxford, UK

**Abstract.** In order to enable interoperability between ontology-based systems, ontology matching techniques have been proposed. However, when the generated mappings suffer from logical flaws, their usefulness may be diminished. In this paper we present a multi-strategy approach to detect and correct violations of the so-called conservativity principle where novel subsumption entailments between named concepts in one of the input ontologies are considered as unwanted. The practical applicability of the proposed approach is experimentally demonstrated on the datasets from the Ontology Alignment Evaluation Initiative (OAEI).

## 1 Introduction

Ontologies play a key role in the development of the Semantic Web and are being used in many diverse application domains, ranging from biomedicine to energy industry. An application domain can be modeled with different points of view and purposes. This situation usually leads to the development of different ontologies that intuitively overlap, but they use different naming and modeling conventions.

The problem of (semi-)automatically computing mappings between independently developed ontologies is usually referred to as the *ontology matching problem*. A number of sophisticated ontology matching systems have been developed in the last years [5, 23]. Ontology matching systems, however, rely on lexical and structural heuristics and the integration of the input ontologies and the mappings may lead to many undesired logical consequences. In [12] three principles were proposed to minimize the number of potentially unintended consequences, namely: *(i) consistency principle*, the mappings should not lead to unsatisfiable classes in the integrated ontology, *(ii) locality principle*, the mappings should link entities that have similar *neighbourhoods*, *(iii) conservativity principle*, the mappings should not introduce new semantic relationships between concepts from one of the input ontologies.

The occurrence of these violations is frequent, even in the reference mapping sets of the Ontology Alignment Evaluation Initiative[3] (*OAEI*). Also manually curated alignments, such as *UMLS-Metathesaurus* [1] (*UMLS*), a comprehensive effort for integrat-

* This work was supported by the EU FP7 IP project Optique (no. 318338) and by the Italian PRIN 2010LHT4KM CINA.

[3] `http://oaei.ontologymatching.org/`

ing biomedical knowledge bases, suffer from these violations. Violations of these principles may hinder the usefulness of ontology mappings [25]. These principles has been actively investigated in the last years (*e.g.*, [17, 9, 12, 10, 16, 21]).

In this paper we combine our previous approach [25] for detecting and solving conservativity principle violations, based on the *assumption of disjointness* [22] and the projection of the input ontologies to Horn propositional logic, with a new one based on Answer Set Programming [24], addressing violations between entities already involved in a subsumption relationship. Our evaluation supports the effectiveness of the combined approach in the detection and correction of an extended notion of conservativity violations, without harming efficiency, even on the largest test cases of *OAEI*.

The remainder of the paper is organised as follows. Section 2 presents the preliminaries. Section 3 introduces our motivating scenario. Section 4 describes our method. Section 5 presents the conducted evaluation. Section 6 provides a comparison with relevant related work. Finally, Section 7 gives some conclusions and future work lines.

## 2 Preliminaries

This section introduces the representation of ontology mappings, the notions of semantic difference and conservativity principle, and the basics on Answer Set Programming.

**Representation of Ontology Mappings.** Mappings are 5-tuples of the form $\langle id, e_1, e_2, n, \rho \rangle$, with $id$ a unique identifier, $e_1, e_2$ entities in the vocabulary or signature of the relevant input ontologies (*i.e.*, $e_1 \in \mathsf{Sig}(\mathcal{O}_1)$ and $e_2 \in \mathsf{Sig}(\mathcal{O}_2)$), $n$ a confidence measure between 0 and 1, and $\rho$ a relation between $e_1$ and $e_2$, typically subsumption, equivalence, or disjointness. Mappings are usually represented as OWL 2 axioms for reusing the currently available OWL 2 reasoning infrastructure, but alternative formal semantics have been proposed (*e.g.*, [2]).

**Semantic Consequences of the Integration.** The ontology resulting from the integration of two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ via a set of mappings $\mathcal{M}$ may entail axioms that do not follow from $\mathcal{O}_1$, $\mathcal{O}_2$, or $\mathcal{M}$ alone. These new semantic consequences can be captured by the notion of *deductive difference* [15].

Intuitively, the deductive difference between $\mathcal{O}$ and $\mathcal{O}'$ w.r.t. a signature $\Sigma$ is the set of entailments constructed over $\Sigma$ that do not hold in $\mathcal{O}$, but do hold in $\mathcal{O}'$. However, no algorithm is available for computing it for DLs more expressive than $\mathcal{EL}$, for which the existence of tractable algorithms is still open [15]

**Definition 1 (Approximation of the Deductive Difference).** *Let $A, B$ be atomic concepts from $\Sigma \cup \{\top, \bot\}$, $\mathcal{O}$ and $\mathcal{O}'$ be two OWL 2 ontologies, with $\Sigma$ is a signature. We define the approximation of the $\Sigma$-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) $\mathcal{O} \not\models A \sqsubseteq B$, and (ii) $\mathcal{O}' \models A \sqsubseteq B$.*

In order to avoid the drawbacks of the deductive difference, in this paper we rely on the *approximation* given in Definition 1. This approximation only requires comparing the classification hierarchies of $\mathcal{O}$ and $\mathcal{O}'$ provided by an OWL 2 reasoner, and it has successfully been used in the past in the context of ontology integration [11].

**Conservativity Principle.** The conservativity principle (as formulated in [12]) states that the integrated ontology $\mathcal{O}_\mathcal{U} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ should not induce any change in the concept hierarchies of the input ontologies. That is, the sets $\mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})$ and $\mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})$ must be empty for signatures $\Sigma_1 = \mathsf{Sig}(\mathcal{O}_1)$ and $\Sigma_2 = \mathsf{Sig}(\mathcal{O}_2)$, respectively. In [25] we proposed a *basic* variant of the conservativity principle where $\mathcal{O}_\mathcal{U}$ is required not to introduce new subsumption relationships between concepts from one of the input ontologies, unless they were already involved in a subsumption relationship or they shared a common descendant. In this paper, in addition to these *basic violations*, we also aim at addressing violations between concepts already involved in a subsumption relationship (*i.e.*, resulting in an equivalence between them), denoted as *equivalence conservativity principle violations*, or simply *equivalence violations*. As in [25], we assume that the mappings $\mathcal{M}$ are coherent w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$ (*i.e.*, $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_1 \cup \mathcal{O}_2, \mathcal{O}_\mathcal{U})$ does not contain any axiom $A \sqsubseteq \bot$, for any $A \in \Sigma = \mathsf{Sig}(\mathcal{O}_1 \cup \mathcal{O}_2)$).

**Definition 2 (Conservativity Principle Violations).** *Let $\mathcal{O}$ be one of the input ontologies and $\mathsf{Sig}(\mathcal{O})$ be its signature, let $\mathcal{O}_\mathcal{U}$ be the integrated ontology, and let $A, B$ be atomic concepts in $\mathsf{Sig}(\mathcal{O}) \cup \{\top, \bot\}$. We define two sets of violations of $\mathcal{O}_\mathcal{U}$ w.r.t. $\mathcal{O}$:*

- *basic violations, denoted $\mathsf{basicViol}(\mathcal{O}, \mathcal{O}_\mathcal{U})$, as the set of $A \sqsubseteq B$ axioms satisfying: (i) $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O})}(\mathcal{O}, \mathcal{O}_\mathcal{U})$, (ii) $\mathcal{O} \not\models B \sqsubseteq A$, and (iii) there is no $C$ in $\mathsf{Sig}(\mathcal{O}) \cup \{\top, \bot\}$ s.t. $\mathcal{O} \models C \sqsubseteq A$, and $\mathcal{O} \models C \sqsubseteq B$.*
- *equivalence violations, denoted $\mathsf{eqViol}(\mathcal{O}, \mathcal{O}_\mathcal{U})$, as the set of $A \equiv B$ axioms satisfying: (i) $\mathcal{O}_\mathcal{U} \models A \equiv B$, (ii) $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O})}(\mathcal{O}, \mathcal{O}_\mathcal{U})$ and/or $B \sqsubseteq A \in \mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O})}(\mathcal{O}, \mathcal{O}_\mathcal{U})$.*

As discussed, for basic violations the *assumption of disjointness* can be applied, that is, if two atomic concepts $A, B$ from one of the input ontologies are not involved in a subsumption relationship nor share a common subconcept (excluding $\bot$) they can be considered as disjoint. Hence, the repair of these violations can be reduced to a consistency repair, if the input ontologies are extended with sufficient disjointness axioms. The same does not necessarily hold for equivalence violations, for which a suitable repair algorithm, based on *ASP*, is introduced (see Section 4.2).

**Answer Set Programming.** *ASP* is a declarative programming language able to express complex search problems up to $\Sigma_2^P$ complexity class (and its complement $\Pi_2^P$ [6]). Each *ASP* program is a set of rules of the form $Head \leftarrow Body$, where $Head$ can be a disjunction of atoms. More formally, a disjunctive rule has the form $a_1 \vee \ldots \vee a_n \leftarrow b_1, \ldots, b_k, not\ b_{k+1}, \ldots, not\ b_m$. Rules with an empty $Body$ (*i.e.*, $k = m = 0$) are called *facts*. *ASP* rules can also include optimization-oriented constructs for minimizing numeric variables or the cardinality of a predicate. These rules are called *soft constraints*, in opposition to *hard constraints*, which are rules with an empty $Head$. In *ASP*, solutions are called *stable models* [6].

## 3 Running Example

In this section, we show an example involving the different kinds of conservativity principle violations arising in the integration of two ontologies modeling knowledge in oil and gas industry [25]. Table 1 shows the relevant fragments of the two ontologies.

**Table 1.** Fragments of the ontologies used in Optique project [25].

| Ontology $\mathcal{O}_1$ | Ontology $\mathcal{O}_2$ |
|---|---|
| $\alpha_1$ WellBore $\sqsubseteq \exists$belongsTo.Well | $\beta_1$ Exploration_well $\sqsubseteq$ Well |
| $\alpha_2$ WellBore $\sqsubseteq \exists$hasOperator.Operator | $\beta_2$ Explor_borehole $\sqsubseteq$ Borehole |
| $\alpha_3$ WellBore $\sqsubseteq \exists$locatedIn.Field | $\beta_3$ Appraisal_exp_borehole $\sqsubseteq$ Explor_borehole |
| $\alpha_4$ AppraisalWellBore $\sqsubseteq$ WellBore | $\beta_4$ Appraisal_well $\sqsubseteq$ Well |
| $\alpha_5$ ExplorationWellBore $\sqsubseteq$ WellBore | $\beta_5$ Field $\sqsubseteq \exists$hasFieldOperator.Field_operator |
| $\alpha_6$ Operator $\sqsubseteq$ Owner | $\beta_6$ Field_operator $\sqcap$ Owner $\sqsubseteq$ Field_owner |
| $\alpha_7$ Operator $\sqsubseteq$ Company | $\beta_7$ Company $\sqsubseteq$ Field_operator |
| $\alpha_8$ Field $\sqsubseteq \exists$hasOperator.Company | $\beta_8$ Field_owner $\sqsubseteq$ Owner |
| $\alpha_9$ Field $\sqsubseteq \exists$hasOwner.Owner | $\beta_9$ Borehole $\sqsubseteq$ Continuant $\sqcup$ Occurrent |

**Table 2.** Ontology mappings for the vocabulary in $\mathcal{O}_1$ and $\mathcal{O}_2$.

| Mappings $\mathcal{M}$ | | | | |
|---|---|---|---|---|
| **id** | $\mathbf{e_1}$ | $\mathbf{e_2}$ | **n** | $\rho$ |
| $m_1$ | $\mathcal{O}_1$:Well | $\mathcal{O}_2$:Well | 0.9 | $\equiv$ |
| $m_2$ | $\mathcal{O}_1$:WellBore | $\mathcal{O}_2$:Borehole | 0.7 | $\equiv$ |
| $m_3$ | $\mathcal{O}_1$:ExplorationWellBore | $\mathcal{O}_2$:Exploration_well | 0.6 | $\sqsubseteq$ |
| $m_4$ | $\mathcal{O}_1$:ExplorationWellBore | $\mathcal{O}_2$:Explor_borehole | 0.8 | $\equiv$ |
| $m_5$ | $\mathcal{O}_1$:AppraisalWellBore | $\mathcal{O}_2$:Appraisal_exp_borehole | 0.7 | $\equiv$ |
| $m_6$ | $\mathcal{O}_1$:Field | $\mathcal{O}_2$:Field | 0.9 | $\equiv$ |
| $m_7$ | $\mathcal{O}_1$:Operator | $\mathcal{O}_2$:Field_operator | 0.7 | $\sqsupseteq$ |
| $m_8$ | $\mathcal{O}_1$:Company | $\mathcal{O}_2$:Company | 0.9 | $\equiv$ |
| $m_9$ | $\mathcal{O}_1$:hasOperator | $\mathcal{O}_2$:hasFieldOperator | 0.6 | $\equiv$ |
| $m_{10}$ | $\mathcal{O}_1$:Owner | $\mathcal{O}_2$:Owner | 0.9 | $\equiv$ |

**Table 3.** Example of conservativity principle violations.

| $\sigma$ | Entailment: | follows from: | basicViol$(\mathcal{O}_i, \mathcal{O}_\mathcal{U})$? | eqViol$(\mathcal{O}_i, \mathcal{O}_\mathcal{U})$? |
|---|---|---|---|---|
| $\sigma_1$ | $\mathcal{O}_2$:Explor_borehole $\sqsubseteq$ $\mathcal{O}_2$:Exploration_well | $m_3, m_4$ | YES | NO |
| $\sigma_2$ | $\mathcal{O}_1$:AppraisalWellBore $\sqsubseteq$ $\mathcal{O}_1$:ExplorationWellBore | $\beta_3, m_4, m_5$ | YES | NO |
| $\sigma_3$ | $\mathcal{O}_2$:Field_operator $\sqsubseteq$ $\mathcal{O}_2$:Field_owner | $\alpha_6, \beta_6, m_7, m_{10}$ | YES | NO |
| $\sigma_4$<br>$\sigma_5$ | $\mathcal{O}_1$:Company $\equiv$ $\mathcal{O}_1$:Operator<br>$\mathcal{O}_2$:Field_operator $\equiv$ $\mathcal{O}_2$:Company | $\alpha_7, \beta_7, m_7, m_8$ | NO | YES |
| $\sigma_6$ | $\mathcal{O}_1$:Company $\sqsubseteq$ $\mathcal{O}_1$:Owner | $\sigma_4, \alpha_6$ | YES | NO |
| $\sigma_7$ | $\mathcal{O}_2$:Company $\sqsubseteq$ $\mathcal{O}_2$:Field_owner | $\sigma_3, \sigma_5$ | YES | NO |

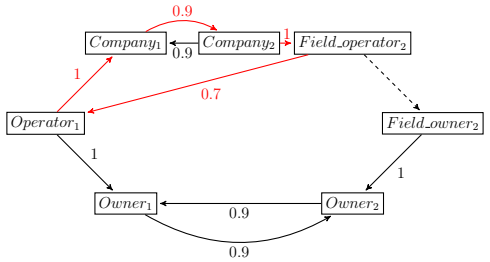


**Fig. 1.** Graph representation of the fragment of the aligned ontology of Tables 1 involved in conservativity violations. Dashed arcs represent inferred axioms, while red arcs are those involved in equivalence violations. Each non-inferred arc is labeled with its confidence value.

Assume that the set of mappings $\mathcal{M}$ in Table 2, between $\mathcal{O}_1$ and $\mathcal{O}_2$, is generated by an off-the-shelf ontology alignment system. As described in Section 2, mappings are represented as 5-tuples; for example the mapping $m_2$ suggests an equivalence relationship between the entities $\mathcal{O}_1$:WellBore and $\mathcal{O}_2$:Borehole, with confidence 0.7.

**Algorithm 1** Algorithm to detect and solve basic violations

**Input:** $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; $\mathcal{M}$: (coherent) input mappings;
**Output:** $\mathcal{M}'$: output mappings; $\mathcal{R}^{\approx}$: approximate repair; $disj$: number of disjointness rules

1: $\langle \mathcal{O}'_1, \mathcal{O}'_2 \rangle := \mathsf{ModuleExtractor}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$
2: $\langle \mathcal{P}_1, \mathcal{P}_2 \rangle := \mathsf{PropositionalEncoding}(\mathcal{O}'_1, \mathcal{O}'_2)$
3: $SI_1 := \mathsf{StructuralIndex}(\mathcal{O}'_1)$, $SI_2 := \mathsf{StructuralIndex}(\mathcal{O}'_2)$
4: $SI_{\mathcal{U}} := \mathsf{StructuralIndex}(\mathcal{O}'_1 \cup \mathcal{O}'_2 \cup \mathcal{M})$
5: $\langle \mathcal{P}_1^d, disj_1 \rangle := \mathsf{DisjointAxiomsExtension}(\mathcal{P}_1, SI_1, SI_{\mathcal{U}})$ ▷ See Algorithm 2
6: $\langle \mathcal{P}_2^d, disj_2 \rangle := \mathsf{DisjointAxiomsExtension}(\mathcal{P}_2, SI_2, SI_{\mathcal{U}})$
7: $\langle \mathcal{M}', \mathcal{R}^{\approx} \rangle := \mathsf{MappingRepair}(\mathcal{P}_1^d, \mathcal{P}_2^d, \mathcal{M})$ ▷ See Algorithm 2 in [13]
8: **return** $\langle \mathcal{M}', \mathcal{R}^{\approx}, disj_1 + disj_2 \rangle$

The integrated ontology $\mathcal{O}_{\mathcal{U}} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$, however, violates the conservativity principle, according to Definition 2, and introduces unwanted subsumption relationships (see Table 3). Note that the entailments $\sigma_4$ and $\sigma_5$ are equivalence violations not belonging to the set of basic violations, since $\mathcal{O}_1$:Company and $\mathcal{O}_1$:Operator (resp. $\mathcal{O}_2$:Field_operator and $\mathcal{O}_2$:Company) are involved in a subsumption relationship in $\mathcal{O}_1$ (resp. $\mathcal{O}_2$). As already discussed, these entailments cannot be repaired by the approach introduced in [25], and in addition they also lead to basic violations ($\sigma_6$ and $\sigma_7$), as shown in Figure 1. Note that equivalence violations $\sigma_4$, $\sigma_5$ can be repaired by the *ASP*-based approach. As shown in [25], any of these conservativity violations may hinder the usefulness of the generated mappings, in a query answering scenario.

# 4 Methods

This section describes the algorithms composing our multi-strategy approach, one addressig basic violations (Section 4.1), the other the equivalence ones (Section 4.2).

## 4.1 Approximate Repair of Basic Conservativity Principle Violations

We have reduced the problem of detecting and solving basic violations, to a mapping (incoherence) repair problem. Currently, our method uses the indexing and reasoning techniques of *LogMap*, an ontology matching and mapping repair system [10, 13].

Algorithm 1 shows the pseudocode of the implemented method. The algorithm accepts as input two OWL 2 ontologies, $\mathcal{O}_1$ and $\mathcal{O}_2$, and a set of mappings $\mathcal{M}$ which are coherent[4] w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$. The problem size is reduced by extracting two locality-based modules [3] ($\mathcal{O}'_1$ and $\mathcal{O}'_2$) using the entities involved in the mappings $\mathcal{M}$ as seed signatures (line 1, Algorithm 1). The output is the number of added disjointness during the process $disj$, a set of mappings $\mathcal{M}'$, and an approximate repair $\mathcal{R}^{\approx}$ s.t. $\mathcal{M}' = \mathcal{M} \setminus \mathcal{R}^{\approx}$. The repair $\mathcal{R}^{\approx}$ aims at solving most of the basic violations of $\mathcal{M}$ w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$. We next describe the techniques used in each step.

**Propositional Horn Encoding.** The modules $\mathcal{O}'_1$ and $\mathcal{O}'_2$ are encoded as the Horn propositional theories, $\mathcal{P}_1$ and $\mathcal{P}_2$ (line 2 in Algorithm 1). The encoding includes rules of the form $A_1 \wedge \ldots \wedge A_n \rightarrow B$, with $A_i, B$ atomic concepts. For example, the concept

[4] Note that $\mathcal{M}$ may be the result of a prior mapping (incoherence) repair process.

hierarchy provided by an OWL 2 reasoner (*e.g.*, [18, 14]) is encoded as $A \rightarrow B$ rules, while explicit disjointness axioms between atomic concepts are encoded as $A_i \wedge A_j \rightarrow \mathsf{false}$. Note that input mappings $\mathcal{M}$ can already be seen as propositional implications.

*Example 1.* Consider the ontologies and mappings in Tables 1 and 2. Axiom $\beta_6$ is encoded as rule $\mathsf{Field\_operator} \wedge \mathsf{Owner} \rightarrow \mathsf{Field\_owner}$, while the mapping $m_2$ is translated into rules $\mathcal{O}_1$:WellBore $\rightarrow \mathcal{O}_2$:Borehole, and $\mathcal{O}_2$:Borehole $\rightarrow \mathcal{O}_1$:WellBore.

**Structural Index.** The concept hierarchies provided by an OWL 2 reasoner (excluding $\bot$) and the explicit disjointness axioms of the modules $\mathcal{O}'_1$ and $\mathcal{O}'_2$ are efficiently indexed using an interval labelling schema (line 3 in Algorithm 1). This structural index allows us to answer many entailment queries over the concept hierarchy as an index lookup operation (*i.e.*, without the need of an OWL 2 reasoner).

**Disjointness Axioms Extension.** In order to reduce the (basic) conservativity problem to a mapping incoherence repair problem, following the notion of *assumption of disjointness*, we need to automatically add sufficient disjointness axioms into each module $\mathcal{O}'_i$. However, additional disjointness axioms $\delta$ may lead to unsatisfiable classes in $\mathcal{O}'_i \cup \delta$.

*Example 2.* Consider the axiom $\beta_9$ from Table 1. Following the *assumption of disjointness* a very naïve algorithm would add disjointness axioms between Borehole, Continuant and Occurrent, which would make Borehole unsatisfiable.

For avoiding an extensive use of a costly OWL 2 reasoner, our method exploits the propositional encoding and structural indexing of the input ontologies. Thus, checking for unsatisfiabilities introduced by candidate disjointness axioms in $\mathcal{O}'_i \cup \delta$ is restricted to the Horn propositional case. We have implemented an algorithm to extend the propositional theories $\mathcal{P}_1$ and $\mathcal{P}_2$ with disjointness rules of the form $A \wedge B \rightarrow \bot$ (see lines 5-6 in Algorithm 1). This algorithm guarantees that, for every propositional variable $A$ in the extended propositional theory $\mathcal{P}^d_i$ (with $i \in \{1, 2\}$), the theory $\mathcal{P}^d_i \cup \{true \rightarrow A\}$ is satisfiable. Note that this does not necessarily hold when considering the OWL 2 ontology modules, $\mathcal{O}'_1$ and $\mathcal{O}'_2$, as discussed above.

*LogMap* provides a structural index $SI$ to check if two propositional variables (*i.e.*, ontological classes) are disjoint ($\mathsf{areDisj}(SI, A, B)$), they keep a sub/super-class relationship ($\mathsf{inSubSupRel}(SI, A, B)$), or they share a descendant ($\mathsf{shareDesc}(SI, A, B)$).

Nonetheless, the massive addition of disjointness rules is, in general, prohibitive for large ontologies [25]. Our key ingredient to achieve scalability is to focus only on the cases where a basic violation occurs in the integrated ontology $\mathcal{O}_\mathcal{U} = \mathcal{O}'_1 \cup \mathcal{O}'_2 \cup \mathcal{M}$, w.r.t. one of the ontology modules $\mathcal{O}'_i$ (with $i \in \{1, 2\}$); *i.e.*, adding a disjointness axiom between each pair of classes $A, B \in \mathcal{O}'_i$ s.t. $A \sqsubseteq B \in \mathsf{basicViol}(\mathcal{O}'_i, \mathcal{O}_\mathcal{U})$, as in Definition 2. Algorithm 2 implements this idea for the Horn propositional case and extensively exploits the structural indexing to identify the basic violations (line 2, Algorithm 2). Note that this algorithm also requires to compute the structural index of the integrated ontology and thus its classification with an OWL 2 reasoner (line 4, Algorithm 1). The classification cost of the integrated ontology is usually much higher than the classification of the input ontologies individually. However, this was not a bottleneck in our experiments (see Section 5).

**Algorithm 2** Disjointness axioms extension

**Input:** $\mathcal{P}$: propositional theory; $SI$: structural index $SI_{\mathcal{U}}$: structural index of the union ontology
**Output:** $\mathcal{P}^d$: extended propositional theory;$disj$: number of disjointness rules

1: $disj := 0, \mathcal{P}^d := \mathcal{P}$
2: **for** $A \rightarrow B \in$ BasicConservativityViolations$(SI, SI_{\mathcal{U}})$ **do**
3: **if not** (areDisj$(SI, A, B)$) **then**
4: $\mathcal{P}^d := \mathcal{P}^d \cup \{A \wedge B \rightarrow false\}$
5: $SI :=$ updateIndex$(SI, A \sqcap B \rightarrow \bot)$
6: $disj := disj + 1$
7: **end if**
8: **end for**
9: **return** $\langle \mathcal{P}^d, disj \rangle$

**Mapping Repair.** Line 7 of Algorithm 1 uses the mapping (incoherence) repair algorithm presented in [13]for the extended Horn propositional theories $\mathcal{P}_1^d$ and $\mathcal{P}_2^d$, and the input mappings $\mathcal{M}$. The mapping repair process exploits the Dowling-Gallier algorithm for propositional Horn satisfiability [4] and checks, for every propositional variable $A \in \mathcal{P}_1^d \cup \mathcal{P}_2^d$, the satisfiability of the propositional theory $\mathcal{P}_A = \mathcal{P}_1^d \cup \mathcal{P}_2^d \cup \mathcal{M} \cup \{true \rightarrow A\}$. Satisfiability of $\mathcal{P}_A$ is checked in worst-case linear time in the size of $\mathcal{P}_A$, and the number of Dowling-Gallier calls is also linear in the number of propositional variables in $\mathcal{P}_1^d \cup \mathcal{P}_2^d$. The algorithm records the *conflicting* mappings involved in the unsatisfiability, which will be considered for the subsequent repair process. The unsatisfiability will be fixed by removing some of the identified mappings. In case of multiple options, mapping confidence will be used as a differentiating factor.[5]

*Example 3.* Consider the propositional encoding $\mathcal{P}_1$ and $\mathcal{P}_2$ of the axioms of Table 1 and the mappings $\mathcal{M}$ in Table 2, seen as propositional rules. $\mathcal{P}_1^d$ and $\mathcal{P}_2^d$ have been created by adding disjointness rules to $\mathcal{P}_1$ and $\mathcal{P}_2$, according to Algorithm 2. For example, $\mathcal{P}_2^d$ includes the rule $\psi = \mathcal{O}_2$:Well $\wedge$ $\mathcal{O}_2$:Borehole $\rightarrow false$. The mapping repair algorithm identifies the propositional theory $\mathcal{P}_1^d \cup \mathcal{P}_2^d \cup \mathcal{M} \cup \{true \rightarrow \mathcal{O}_1$:ExplorationWellbore$\}$ as unsatisfiable. This is due to the combination of the mappings $m_3$ and $m_4$, the propositional projection of axioms $\beta_1$ and $\beta_2$, and the rule $\psi$. The mapping repair algorithm also identifies $m_3$ and $m_4$ as the cause of the unsatisfiability, and discards $m_3$ since its confidence is smaller than that of $m_4$ (see Table 2).

Algorithm 1 gives as output the number of added disjointness rules $disj$, a set of mappings $\mathcal{M}'$, and an (approximate) repair $\mathcal{R}^{\approx}$ such that $\mathcal{M}' = \mathcal{M} \setminus \mathcal{R}^{\approx}$. $\mathcal{M}'$ is coherent w.r.t. $\mathcal{P}_1^d$ and $\mathcal{P}_2^d$. Furthermore, the propositional theory $\mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{M}'$ does not contain any basic violation w.r.t. $\mathcal{P}_1$ and $\mathcal{P}_2$ (according to the propositional case of Definition 2). However, our incomplete encoding cannot guarantee that $\mathcal{O}_1' \cup \mathcal{O}_2' \cup \mathcal{M}'$ does not contain basic violations w.r.t. $\mathcal{O}_1'$ and $\mathcal{O}_2'$. Nonetheless, our evaluation suggests that the number of remaining violations after repair is typically small (see Section 5).

### 4.2 Repair of Equivalence Conservativity Principle Violations

In this section we describe CycleBreaker algorithm that detects equivalence violations at the taxonomical level, and computes a minimal repair by means of a logic program [6].

[5] Note that the locality principle can be used to compute fresh confidence values, if needed [12].
[6] Formal definitions and proofs are provided in an accompanying technical report [24].

**Algorithm 3** CycleBreaker Algorithm

**Input:** Ontology $O_1$, Ontology $O_2$, Alignment $\mathcal{M}$
**Output:** Repair $\Delta$
1: $O^{\mathcal{M}} = alignOnto(O_1, O_2, \mathcal{M})$
2: $G = createDigraph(O^{\mathcal{M}})$
3: $SCCS_i = tarjan(G, O_i)$, $globalSCCs = tarjan(G)$
4: **for** $S \in globalSCCs$ s.t. $\neg \bigwedge_{i=1}^{2} \Pi_{O_i}(S) \in SCCs_i$ **do**
5: $\quad \Delta \leftarrow \Delta \cup ASP(S)$
6: **end for**
7: **return** $\Delta$

**Algorithm Description.** Algorithm 3 takes as input two ontologies $O_1$,$O_2$, and an alignment $\mathcal{M}$ between them. The aligned ontology $O^{\mathcal{M}}$ is computed (line 1), and $createDigraph$ function (line 2) builds its graph representation, having its named concepts as the set of vertices, and the subsumption axioms between them as (weighted) arcs. Given that the aligned ontology is equivalent, here, to the classification of the input ontologies plus the mappings, only inferred arcs between concepts of the same input ontology may exist. Given that at least one cycle containing all the elements of a *strongly connected components* (SCC) exists, cycle detection for a graph $G'$ can be reduced to computing $SCC(G')$, the set of its SCCs. *Tarjan's* algorithm [26] computes the SCCs of the graph representations of the input ontologies (named local SCCs) and of the aligned one (line 3). Note that not all the cycles leads to an equivalence violation. We therefore distinguish between unsafe and safe cycles as those producing or not a violation. For detecting all the unsafe cycles, it is sufficient to detect the problematic SCCs, identified by the fact that at least one of the two projections on the input ontologies is not a local SCC (line 4). The *ASP* program of Listing 1.1 then computes a minimal repair for each problematic SCC (line 5).

**Listing 1.1.** *ASP* program computing a minimal repair for a problematic SCC.

```
r0:#domain vtx(X,O). #domain vtx(Y,P). #domain vtx(Z,Q).
r1:reachesSafe(X,Y) :- edge(X,Y,C,0), O=P, X!=Y.
r2:reachesSafe(X,Z) :- reachesSafe(X,Y), edge(Y,Z,C,0), O=P, P=Q, X!=Y, Y!=Z, X!=Z.
r3:reaches(X,Y) :- edge(X,Y,C,M), unremoved(edge(X,Y,C,M)), X!=Y.
r4:reaches(X,Z) :- reaches(X,Y), edge(Y,Z,C,M), unremoved(edge(Y,Z,C,M)), X!=Y,
     Y!=Z, X!=Z.
r5:unremoved(edge(X,Y,C,M)) | removed(edge(X,Y,C,M)) :- edge(X,Y,C,M), X!=Y.
r6:unremoved(edge(X,Y,C,0)) :- edge(X,Y,C,0), X!=Y, O=P.
r7:unsafeCycle(Y) :- not reachesSafe(Y,X), reaches(Y,X), reaches(X,Y), O=P, X!=Y.
r8: :- unsafeCycle(Y).
r9:#minimize [ removed(edge(X,Y,C,1)) = C ].
```

(r0) states that $X, Y, Z$ are always vertices, (r1,r2) define transitive predicate $reachesSafe$ expressing vertex reachability in the input ontologies (in isolation), (r3,r4) define transitive predicate $reaches$ expressing vertex reachability in the aligned ontology (*i.e.*, considering edges and unremoved mappings only), (r5) generates models, (r6) allows mapping removal only, (r7) computes unsafe cycles in the graph, (r8) is a hard constraint forbidding the existence of unsafe cycles (r9) is a soft constraint that selects one of the valid models having minimal total weight of removed mappings.

**Repair Computation Using *ASP*.** The input facts for the *ASP* program are the following kinds: (i) predicate $vtx(X, O)$ encodes vertices, where $X$ is a unique vertex id (*e.g.*, vertex label) and $O \in \{1, 2\}$ is the index of the input ontology of the concept, (ii)

**Algorithm 4** Conducted evaluation over the Optique and OAEI data sets

**Input:** $\mathcal{O}_1, \mathcal{O}_2$: input ontologies $\mathcal{M}$: reference mappings for $\mathcal{O}_1$ and $\mathcal{O}_2$

1: $\mathcal{O}_\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$
2: Store size of $\mathsf{Sig}(\mathcal{O}_1)$ (**I**), $\mathsf{Sig}(\mathcal{O}_2)$ (**II**) and $\mathcal{M}$ (**III**)
Compute number of conservativity principle violations:
3: $\mathsf{basicViol} := |\mathsf{basicViol}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})| + |\mathsf{basicViol}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})|$ (**IV**) ▷ basic violations, as in Definition 2
4: $\mathsf{diff}^{\approx} := |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_1)}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})| + |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_2)}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})|$ (**V**) ▷ general notion as in Section 2
5: $\mathsf{eqViol} := |\mathsf{eqViol}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})| + |\mathsf{eqViol}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})|$ (**VI**) ▷ equivalence violations, as in Definition 2
6: Compute a repair $\mathcal{R}^{\approx}$ using Algorithm 1 for $\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}$
7: Store number of added disjointness $disj$ (**VII**), time to compute disjointness rules $\mathrm{t}_d$ (**VIII**)
8: Compute a repair $\mathcal{R}^{SCC}$ using Algorithm 3 for $\mathcal{O}_1, \mathcal{O}_2, \mathcal{M} \setminus \mathcal{R}^{\approx}$
9: Store size global time to compute the combined mapping repair $\mathrm{t}_r$ (**IX**) and size of repair $|\mathcal{R}^{SCC}|$ (**X**)
10: $\mathcal{O}_\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M} \setminus \mathcal{R}^{SCC}$
Compute number of remaining conservativity principle violations:
11: $\mathsf{basicViol} := |\mathsf{basicViol}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})| + |\mathsf{basicViol}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})|$ (**XI**) ▷ basic violations
12: $\mathsf{diff}^{\approx} := |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_1)}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})| + |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_2)}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})|$ (**XII**) ▷ general notion
13: $\mathsf{eqViol} := |\mathsf{eqViol}(\mathcal{O}_1, \mathcal{O}_\mathcal{U})| + |\mathsf{eqViol}(\mathcal{O}_2, \mathcal{O}_\mathcal{U})|$ (**XIII**) ▷ equivalence violations

predicate $edge(X, Y, C, M)$ encodes arcs, where $X$ and $Y$ are vertices, $C$ is an integer encoding for arc weight in $[0 \dots 100]$, $M$ is a boolean flag for mappings.

*Example 4.* The model (*i.e.*, the computed repair) for the logic program on the following facts, encoding the problematic SCC of Figure 1, is equal to mapping $m_7$ of Table 2:

```
vtx(Company1,1). vtx(Operator1,1). vtx(Company2,2). vtx(Field_operator2,2).
edge(Company1,Company2,90,1). edge(Company2,Company1,90,1).
edge(Company2,Field_operator2,100,0). edge(Field_operator2,Operator1,70,1).
```

## 5 Evaluation

In order to evaluate the practical feasibility of our approach, we have conducted the evaluation in Algorithm 4 (Roman numbers refer to stored measurements) over the Optique's use case [25] and the ontologies and reference mapping sets of *OAEI 2013*.[7]

Table 4 shows the size of the evaluated ontologies and mappings (**I**, **II** and **III**). For the Conference dataset we have selected only 5 pair of ontologies for which the reference mappings lead to more than five conservativity principle violations. Note that we count equivalence mappings as two subsumption mappings, and hence $\mathcal{M}$ represents subsumption mappings. Table 4 also shows the conservativity principle violations for the reference mappings (**IV**, **V** and **VI**). For LargeBio and Library the number is expecially large using both the basic variant and the general notion of the conservativity principle.[8] We have run the experiments shown in Table 5 on a desktop computer with an *AMD Fusion A6-3670K* CPU and allocating 12 GB of RAM. The prototype uses *Clingo* 3.0.5[9] *ASP* solver. The obtained results[10] can be summarised as follows:

[7] More information can be found in `http://oaei.ontologymatching.org/2013/`

[8] No OWL 2 reasoner could classify the integrated SNOMED-NCI ontology. The OWL 2 EL reasoner ELK [14] computed a lower bound on the number of conservativity violations.

[9] `http://potassco.sourceforge.net/`

[10] The computation times of lines 1-3 in Algorithm 1 were negligible with respect to the repair and disjointness addition times ($\mathrm{t}_r$ and $\mathrm{t}_d$) and thus they were not included in the result tables.

**Table 4.** Test cases and violations with original reference mappings.

| Dataset | $\mathcal{O}_1 \sim \mathcal{O}_2$ | Problem size | | | Original Violations | | |
|---|---|---|---|---|---|---|---|
| | | I | II | III | IV | V | VI |
| | | $\lvert\mathsf{Sig}(\mathcal{O}_1)\rvert$ | $\lvert\mathsf{Sig}(\mathcal{O}_2)\rvert$ | $\lvert\mathcal{M}\rvert$ | basicViol | diff$^{\approx}$ | eqViol |
| **Optique** | **NPD∼BootsOnto** | 757 | 40,671 | 102 | 214 | 220 | 25 |
| **LargeBio** | **SNOMED∼NCI** | 51,179 | 24,040 | 36,405 | >525,515 | >546,181 | >6,090 |
| | **FMA∼SNOMED** | 10,181 | 13,430 | 17,212 | 125,232 | 127,668 | 1,091 |
| | **FMA∼NCI** | 3,720 | 6,551 | 5,821 | 19,740 | 19,799 | 427 |
| **Anatomy** | **MO∼NCI$_{\text{Anat}}$** | 2,747 | 3,306 | 3,032 | 1,321 | 1,335 | 26 |
| **Library** | **STW∼TheSoz** | 6,575 | 8,376 | 6,322 | 42,045 | 42,872 | 895 |
| **Conference** | **cmt∼confof** | 89 | 75 | 32 | 11 | 11 | 0 |
| | **conference∼edas** | 124 | 154 | 34 | 8 | 8 | 0 |
| | **conference∼iasted** | 124 | 182 | 28 | 9 | 9 | 0 |
| | **confof∼ekaw** | 75 | 107 | 40 | 6 | 6 | 1 |
| | **edas∼iasted** | 154 | 182 | 38 | 7 | 7 | 0 |

**Table 5.** Results of our method to detect and solve conservativity principle violations.

| Dataset | $\mathcal{O}_1 \sim \mathcal{O}_2$ | Solution size | | Times | | Remaining Violations | | |
|---|---|---|---|---|---|---|---|---|
| | | VII | X | VIII | IX | XI | XII | XIII |
| | | #disj | $\lvert\mathcal{R}^{\mathcal{SCC}}\rvert$ | $t_d$(s) | $t_r$(s) | basicViol | diff$^{\approx}$ | eqViol |
| **Optique** | **NPD∼BootsOnto** | 214 | 44 | 2.22 | 1.94 | 0 | 0 | 0 |
| **LargeBio** | **SNOMED∼NCI** | 525,515 | 16,180 | 80 | 3,721 | >51 | >402 | >2 |
| | **FMA∼SNOMED** | 125,232 | 8,369 | 30 | 269 | 0 | 33 | 0 |
| | **FMA∼NCI** | 19,740 | 2,179 | 36 | 36 | 103 | 104 | 0 |
| **Anatomy** | **MO∼NCI$_{\text{Anat}}$** | 1,321 | 493 | 1.46 | 1.34 | 0 | 2 | 0 |
| **Library** | **STW∼TheSoz** | 42,045 | 3,068 | 4.96 | 44 | 0 | 15 | 0 |
| **Conference** | **cmt∼confof** | 11 | 6 | 0.04 | 0.04 | 0 | 0 | 0 |
| | **conference∼edas** | 8 | 6 | 0.07 | 0.07 | 0 | 0 | 0 |
| | **conference∼iasted** | 9 | 3 | 0.21 | 0.14 | 0 | 0 | 0 |
| | **confof∼ekaw** | 6 | 5 | 0.04 | 0.03 | 0 | 0 | 0 |
| | **edas∼iasted** | 7 | 4 | 0.21 | 0.16 | 1 | 1 | 0 |

i The number of added disjointness rules $disj$ (**VII**), as expected, coincides with the number of basic violations, that is computed in a reasonable time also for large testcases (it requires only 80 seconds to compute them in the SNOMED-NCI case).

ii The computed repair $\mathcal{R}^{\mathcal{SCC}}$ (**X**) is rather aggressive and it can remove from 16% (Anatomy) up to 48% (Optique) of the mappings. However, we follow a *better safe than sorry* approach, and we prefer to remove as many violations as possible, rather than preserving potentially conflicting mapping sets.

iii Global repair time $t_r$ (**IX**) is small and it does not represent a bottleneck in spite of the large number of added disjointness rules and multiple applied techniques.

iv The basic violations (**XI**) are fully removed in the Optique, Anatomy and Library cases, and almost fully removed in the Conference and LargeBio ones.

v The number of missed violations is only slightly higher when considering the general notion of the conservativity principle (**XII**), which suggests that basic variant is suitable in practice. Furthermore, these violations are also almost always removed.

vi Restricting to equivalence violations, they are almost completely removed by the combined approach (**XIII**). In order to evaluate the effectiveness of CycleBreaker, we also measured the number of unsolved equivalence violations after applying this repair algorithm, in isolation, on the input reference alignment. With the exception of SNOMED-NCI (failure rate of $0.9\%$), the violations are totally repaired.

## 6 Related Work

The conservativity principle, although indirectly, has been actively studied in the literature. For example, the assumption of disjointness was originally introduced by Schlobach [22] to enhance the repair of ontologies that were underspecified in terms of disjointness axioms. As in our case, also [27, 17, 7], have focused on the addition of a small set of disjointness axioms, for supporting large ontologies.

Ontology matching systems have also dealt with the conservativity principle in order to improve the precision (w.r.t. a reference alignment) of the computed mappings. For example, *ASMOV* [9], *Lily* [28] and *YAM++* [19] implements different heuristics and patterns to minimise the violations of the conservativity principle. *Lily*, in particular, supports an incomplete detection of the equivalence violations, but lacks of a repair technique. Our basic method solves conservativity violations by reducing the problem to a consistency principle violation problem. Concretely, we have extended *LogMap* matcher [10] but other mapping repair systems could be considered (*e.g.*, *Alcomo* [16] or *AML* [21]). Note that these systems only focused on the consistency principle.

The work presented in [8] follows an opposite approach with respect to ours, and considers conservativity violations as false positives, caused by the potential incompleteness of the input ontologies. Hence, the correction strategy, instead of removing mappings, aims at inserting subsumption axioms to the input ontologies, to enrich their concept hierarchies. Authors in [20] also suggest that fixing the input ontologies, in a mapping repair process, could be an alternative to mapping removal.

## 7 Conclusions and Future Work

In this paper we have presented a fully-automatic multi-strategy method to detect and correct conservativity principle violations in practice. We have extended the detect and repair algorithm for basic violations [25] with a repair algorithm based on logic programming, tailored for equivalence violations. The conducted evaluation supports the practical effectivity of our incomplete method. We plan to extend the approach while keeping the current scalability properties. The proposed algorithms follow a "better safe than sorry" approach, suitable for scenarios where the input ontologies are not modifiable (*e.g.*, fully automatic ontology matching systems). Nevertheless we do not discard to explore alternative methods to address the conservativity principle violations. We also plan to involve domain experts in the assessment of the additional disjointness [7], and to suggest extensions to the input ontologies [8] for violations recognised as false positives. We will also evaluate the mappings computed by systems participating at *OAEI*, and the integration of our techniques into *LogMap* matcher.

## References


1. Bodenreider, O.: The Unified Medical Language System (UMLS): integrating biomedical terminology. Nucleic Acids Research 32, 267–270 (2004)
2. Borgida, A., Serafini, L.: Distributed Description Logics: Assimilating Information from Peer Sources. J. Data Sem. 1, 153–184 (2003)

3. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular Reuse of Ontologies: Theory and Practice. J. Artif. Intell. Res. 31, 273–318 (2008)
4. Dowling, W.F., Gallier, J.H.: Linear-Time Algorithms for Testing the Satisfiability of Propositional Horn Formulae. J. Log. Prog. 1(3), 267–284 (1984)
5. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology Alignment Evaluation Initiative: Six Years of Experience. J. Data Sem. 15, 158–192 (2011)
6. Faber, W.: Answer Set Programming. In: Reasoning Web. pp. 162–193 (2013)
7. Ferré, S., Rudolph, S.: Advocatus Diaboli - Exploratory Enrichment of Ontologies with Negative Constraints. In: Int'l Conf. on Knowl. Eng. (EKAW). pp. 42–56 (2012)
8. Ivanova, V., Lambrix, P.: A Unified Approach for Aligning Taxonomies and Debugging Taxonomies and their Alignments. In: Eur. Sem. Web Conf. (ESWC), pp. 1–15. Springer (2013)
9. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology Matching With Semantic Verification. J. Web Sem. 7(3), 235–251 (2009)
10. Jiménez-Ruiz, E., Cuenca Grau, B.: LogMap: Logic-based and Scalable Ontology Matching. In: Int'l Sem. Web Conf. (ISWC). pp. 273–288 (2011)
11. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Ontology integration using mappings: Towards getting the right logical consequences. In: Eur. Sem. Web Conf. (2009)
12. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Logic-based Assessment of the Compatibility of UMLS Ontology Sources. J. Biomed. Semant. 2(Suppl 1), S2 (2011)
13. Jiménez-Ruiz, E., Meilicke, C., Cuenca Grau, B., Horrocks, I.: Evaluating Mapping Repair Systems with Large Biomedical Ontologies. In: Description Logics. pp. 246–257 (2013)
14. Kazakov, Y., Krötzsch, M., Simancik, F.: Concurrent Classification of EL Ontologies. In: Int'l Sem. Web Conf. (ISWC). pp. 305–320 (2011)
15. Konev, B., Walther, D., Wolter, F.: The Logical Difference Problem for Description Logic Terminologies. In: Int'l Joint Conf. on Automated Reasoning (IJCAR). pp. 259–274 (2008)
16. Meilicke, C.: Alignments Incoherency in Ontology Matching. Ph.D. thesis (2011)
17. Meilicke, C., Völker, J., Stuckenschmidt, H.: Learning Disjointness for Debugging Mappings between Lightweight Ontologies. In: Int'l Conf. on Knowl. Eng. (EKAW). pp. 93–108 (2008)
18. Motik, B., Shearer, R., Horrocks, I.: Hypertableau Reasoning for Description Logics. J. Artif. Intell. Res. (JAIR) 36, 165–228 (2009)
19. Ngo, D., Bellahsene, Z.: YAM++ : A Multi-strategy Based Approach for Ontology Matching Task. In: Int'l Conf. on Knowl. Eng. (EKAW). pp. 421–425 (2012)
20. Pesquita, C., Faria, D., Santos, E., Couto, F.M.: To repair or not to repair: reconciling correctness and coherence in ontology reference alignments. In: Ontology Matching (OM) (2013)
21. Santos, E., Faria, D., Pesquita, C., Couto, F.: Ontology Alignment Repair Through Modularization and Confidence-based Heuristics. arXiv:1307.5322 preprint (2013)
22. Schlobach, S.: Debugging and Semantic Clarification by Pinpointing. In: Eur. Sem. Web Conf. (ESWC), pp. 226–240. Springer (2005)
23. Shvaiko, P., Euzenat, J.: Ontology Matching: State of the Art and Future Challenges. IEEE Transactions on Knowl. and Data Eng. (TKDE) (2012)
24. Solimando, A., Guerrini, G.: Coping with Conservativity Principle Violations in Ontology Mappings. Tech. Rep. DIBRIS-TR-14-01, University of Genova (Jan 2014), `ftp://ftp.disi.unige.it/person/SolimandoA/cycleMappingDbgExt.pdf`
25. Solimando, A., Jiménez-Ruiz, E., Guerrini, G.: Detecting and Correcting Conservativity Principle Violations in Ontology-to-Ontology Mappings. In: Int'l Sem. Web Conf. (2014), `ftp://ftp.disi.unige.it/person/SolimandoA/conservLogMap.pdf`
26. Tarjan, R.: Depth-first Search and Linear Graph Algorithms. SIAM J. Comp. 1(2) (1972)
27. Völker, J., Vrandecic, D., Sure, Y., Hotho, A.: Learning Disjointness. In: Eur. Sem. Web Conf. (ESWC). pp. 175–189 (2007)
28. Wang, P., Xu, B.: Debugging Ontology Mappings: A Static Approach. Computing and Informatics 27(1), 21–36 (2012)