Knowl Inf Syst (2017) 51:775–819
DOI 10.1007/s10115-016-0983-3

REGULAR PAPER

Minimizing conservativity violations in ontology alignments: algorithms and evaluation

Alessandro Solimando[1] · **Ernesto Jiménez-Ruiz**[2] · **Giovanna Guerrini**[1]

Received: 20 January 2016 / Revised: 28 July 2016 / Accepted: 17 August 2016 /
Published online: 27 August 2016

Abstract In order to enable interoperability between ontology-based systems, ontology matching techniques have been proposed. However, when the generated mappings lead to undesired logical consequences, their usefulness may be diminished. In this paper, we present an approach to detect and minimize the 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.

Keywords Ontology alignment · Ontology matching · Ontology alignment debugging · Mapping repair

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 may have been modelled 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 modelling 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 [14,71]. 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.

✉ Alessandro Solimando
alessandro.solimando@unige.it

[1] DIBRIS, Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi, University of Genova, Via Dodecaneso 35, 16146 Genova, Italy

[2] Department of Computer Science, University of Oxford, Oxford, UK

In [36], three principles were proposed to minimize the number of potentially unintended consequences, namely (i) consistency principle: the mappings should not lead to unsatisfiable concepts in the integrated ontology; (ii) conservativity principle: the mappings should not introduce new semantic relationships between concepts from one of the input ontologies; and (iii) locality principle: the mappings should link entities that have similar *neighbourhoods*.

These alignment principles have been actively investigated in the last years (e.g., [32,33,36,54,56,57,66]). Violations to these principles are frequent, even in the reference mapping sets and the alignments generated by the best-performing matchers of the Ontology Alignment Evaluation Initiative[1] (*OAEI*). Also manually curated alignments, such as the *UMLS Metathesaurus* [5] (*UMLS*),[2] a comprehensive effort for integrating biomedical knowledge bases, suffer from these violations [36]. The occurrence of these violations may hinder the usefulness of ontology mappings. The practical effect of these violations is clearly evident when ontology alignments are involved in complex tasks such as query answering [54,78]. The undesired logical consequences caused by violations can either prevent query answering or cause incorrect results. In order to reduce existing violations, alignment repair methods typically remove a subset of the alignment, given that input ontologies are considered as immutable, a common setting in ontology alignment repair scenarios.

However, the different nature of the alignment principles should be noted. Violations of the consistency principle, unlike violations of the conservativity and locality principles, always lead to an undesired logical consequence (i.e., unsatisfiability of a concept) and they should always be avoided. Conservativity and locality violations may also lead to undesired logical consequences; however, they may also represent false positives and reveal incompleteness in one of the input ontologies. In Sect. 8, we discuss alternative approaches that suggest to fix the input ontologies instead of repairing the alignment (e.g., [48,62]).

In this paper, we focus on the conservativity violations and we follow a “better safe than sorry” approach (i.e., we treat violations as undesired consequences led by the mappings). Conservativity violations are presented in two flavours, namely *subsumption violations* and *equivalence violations*. The (potential) challenging number of conservativity violations requires to exploit the intrinsic characteristics of these two flavours that result in the development of different approaches for their repair. The detection and correction of subsumption violations relies on the *assumption of disjointness* [67] and it is reduced to a consistency principle violation problem, while equivalence violations are addressed using a combination of graph theory and logic programming. These two methods are combined into a *multi-strategy* approach addressing both types of violations. Our extensive evaluation supports the effectiveness of the individual and combined approaches in the detection and correction of conservativity violations.

The present paper extends [74,75] under the following aspects: all the experimental evaluations provided here cover both reference alignments and alignments computed by participating systems of the *OAEI* 2012–2014 campaigns, where previous papers covered only the reference alignments of the *OAEI*. Compared to [75], the present article fully details the proposed method, including a correctness proof of the technique for adding disjointness clauses to Horn propositional formulas, on which our technique heavily relies. Furthermore, [75] only dealt with the subsumption violations flavour, while in this paper we also cover in detail the equivalence violations flavour. Concerning [74],[3] all the technical details and proofs are now provided. In addition, the results of the evaluation of the two possible variants

[1] http://oaei.ontologymatching.org/.

[2] Alignments from UMLS are extracted according to the method defined in [36].

[3] This paper was presented in a workshop without formal proceedings.

of our combined repair approach are now analysed, as well as the results for the independent techniques in isolation, that can be used as baseline results. Finally, an empirical assessment of the impact of our repair methods on the alignment quality (in terms of precision, recall, and f-measure) is now provided.

The remainder of the paper is organized as follows. Section 2 summarizes the basic concepts and definitions we will rely on along the paper. In Sect. 3, we introduce our motivating scenario. Section 4 formally states the problem of computing repairs for equivalence violations and presents an algorithm to solve such violations. Section 5 describes the method and algorithm to solve subsumption violations. Section 6 details additional properties of the proposed methods. In Sect. 7, we present the conducted evaluation. A comparison with relevant related work is provided in Sect. 8. Finally, Sect. 9 gives some conclusions and future work lines.

2 Preliminaries

In this section, we provide the necessary definitions and notions that will be used in the subsequent sections. Section 2.1 briefly introduces OWL 2 and the main elements in an ontology. In Sect. 2.2, we give a formal definition of ontology mapping and ontology alignment (adapted from [15]) with their semantics. In Sect. 2.3, we precisely define the semantic consequences imposed by ontology alignments and we formalize the *consistency* and *conservativity* principles. Finally, Sect. 2.4 covers the necessary preliminaries about graph theory.

2.1 Ontologies and OWL 2

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. The most widely used ontology modelling language is the OWL 2 Web Ontology Language [27], which is a World Wide Web Consortium (W3C) recommendation [84]. Description Logics (DL) are the formal underpinning of OWL 2 [3,30].

An OWL 2 ontology $\mathcal{O}$ is equipped with a signature $Sig(\mathcal{O})$ that is a vocabulary of legal names for the *entities* appearing in the ontology. $Sig(\mathcal{O})$ is composed by the disjoint union of four finite sets: (i) N_C, a set of unary symbols called *named concepts*; (ii) N_R, a set of binary symbols called *named object properties*; (iii) N_D, a set of binary symbols called *data properties*; and (iv) N_I, a set of constant symbols called *named individuals*.

OWL 2 ontologies can be seen as a set of axioms that are conformant to the syntactic rules and constraints imposed by their underlying DL language [30] and built using the elements of the signature. The *classification* of $\mathcal{O}$, denoted as $\mathcal{C}l(\mathcal{O})$, corresponds to the result of the computation, performed using an OWL 2 reasoner, of the full subsumption/subconcept relation between its named concepts (i.e., elements of N_C). Classification is therefore the subset of the logical closure of an ontology $\mathcal{O}$ s.t. each axiom is of the form $A \sqsubseteq B$, where $A, B \in N_C(\mathcal{O})$ and $\mathcal{O} \models A \sqsubseteq B$.

2.2 Ontology mappings and alignments

Ontology mappings In Definition 2.1, we provide the definition of ontology *mapping* (also called *match* or *correspondence*).

Definition 2.1 Consider two input ontologies $\mathcal{O}_1, \mathcal{O}_2$ and their respective signature $Sig(\mathcal{O}_1)$ and $Sig(\mathcal{O}_2)$. A *mapping* between entities of $\mathcal{O}_1, \mathcal{O}_2$ is a 4-tuple $\langle e, e', r, c\rangle$ such that $e \in Sig(\mathcal{O}_1)$ and $e' \in Sig(\mathcal{O}_2)$, $r \in \{\sqsubseteq, \sqsupseteq, \equiv\}$[4] is a semantic relation, and c is a confidence value. Usually, the real number unit interval $(0 \ldots 1]$ is employed for representing confidence values. Mapping confidence intuitively reflects how reliable a mapping is (i.e., 1 = very reliable, 0 = not reliable).

Ontology alignment Definition 2.2 introduces the notion of *alignment*.

Definition 2.2 An *alignment* $\mathcal{M}$ between two ontologies, namely $\mathcal{O}_1, \mathcal{O}_2$, is a set of mappings between $\mathcal{O}_1$ and $\mathcal{O}_2$.

The main format to represent mappings has been proposed in the context of the *Alignment API*, and it is called RDF Alignment [9]. This format is the standard for the well-known OAEI campaign. In addition, mappings are also represented as standard subclass and equivalence DL axioms. When mappings are expressed through OWL 2 axioms, confidence values are represented as OWL 2 axiom annotations [35]. The representation through standard OWL 2 axioms enables the reuse of the extensive range of OWL 2 reasoning infrastructure that is currently available. We adopt this representation, and in the remainder of the paper, we consider alignments as set of OWL 2 axioms.

Definition 2.3 introduces the notion of *aligned ontology*, resulting from the integration of two input ontologies, through an alignment between them.

Definition 2.3 Let $\mathcal{O}_1, \mathcal{O}_2$ be two (input) ontologies, and let $\mathcal{M}$ be an alignment between them. The ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ is called the *aligned ontology* w.r.t. $\mathcal{O}_1, \mathcal{O}_2$, and $\mathcal{M}$.

$\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ is simply called the aligned ontology when no confusion arises. Note that we assume that the signature of the aligned ontology is always the union of the signatures of the input ontologies. When the input ontologies are clear from the context, we employ the abbreviated notation $\mathcal{O}^{\mathcal{M}}$.

Given that each mapping is translated into an OWL 2 axiom, the aligned ontology is again an OWL 2 ontology. Note that alternative formal semantics for ontology mappings have been proposed in the literature, such as those proposed by Zimmermann et al. in [87] and the semantics associated with the so-called *bridge rules*, in the context of distributed description logics [6,55].

2.3 Semantics of the integration and principles for ontology alignments

This section introduces the semantics of the integration and provides a formal characterization of the consistency and conservativity principles in ontology alignment.

Semantic consequences of the integration The ontology resulting from the integration of two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ via an alignment $\mathcal{M}$ 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* [46,47].

Intuitively, the deductive difference between $\mathcal{O}$ and $\mathcal{O}'$, w.r.t. a signature Σ, is the set of entailments constructed over Σ that do not hold in $\mathcal{O}$, but do hold in $\mathcal{O}'$. The notion of

[4] We exclude disjointness from the semantic relations given that most of the available systems do not compute this relation. Negative constraints are typically harder to identify and assess than positive ones [18].

deductive difference, however, has several drawbacks in practice. First, there is no algorithm available for computing it for DLs more expressive than $\mathcal{EL}$, for which the problem is already *EXPTIME*-complete [52]. In addition, the problem is undecidable for DLs as expressive as $\mathcal{ALCQIO}$[5] [51]. Second, the huge (possibly infinite) number of entailments in the difference is likely to overwhelm users.

In order to avoid the drawbacks of the deductive difference, we rely on the *approximation of the deductive difference*, specified in Definition 2.4.

Definition 2.4 Let A, B be named concepts (including $\top, \bot$), Σ be a signature, $\mathcal{O}$ and $\mathcal{O}'$ be two ontologies. We define 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) $\mathcal{O}' \models A \sqsubseteq B$.

The proposed approximation only requires to compare the classification hierarchies of ontologies $\mathcal{O}$ and $\mathcal{O}'$ (i.e., $\mathcal{C}l(\mathcal{O})$ and $\mathcal{C}l(\mathcal{O}')$). In this paper, we rely on this approximation, which has successfully been used in the past in the context of ontology integration to help users understanding the semantic consequences of this operation [35].

Consistency principle violations Consistency violations evidence either a problem in the mappings or an incompatibility between the input ontologies, because they always result in incoherent and/or inconsistent (aligned) ontologies (i.e., an ontology containing unsatisfiable concepts). The consistency principle is the most widely investigated in the literature, where tools for detecting and automatically repair mappings leading to logical inconsistencies in the aligned ontology have been proposed (e.g., [33,54]). Violations of the consistency principle are (typically) easy to detect with standard reasoning services. However, repairing such violations through standard reasoning services leads to intractability for medium size ontologies and alignments [34,76,77].

The consistency principle requires all the (named) concepts of the aligned ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ to be satisfiable, assuming the union of the input ontologies $\mathcal{O}^{\emptyset}_{\mathcal{O}_1,\mathcal{O}_2} = \mathcal{O}_1 \cup \mathcal{O}_2$ (without the alignment $\mathcal{M}$) does not contain unsatisfiable concepts. In Definition 2.5, we formally define (violations of) the consistency principle.

Definition 2.5 An alignment $\mathcal{M}$ violates the *consistency principle* (i.e., it is incoherent) w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$ if $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}^{\emptyset}_{\mathcal{O}_1,\mathcal{O}_2}, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ contains axioms of the form $A \sqsubseteq \bot$, for all $A \in \Sigma = Sig(\mathcal{O}_1) \cup Sig(\mathcal{O}_2)$. Violations of the consistency principle result in an incoherent aligned ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$.

Conservativity principle violations The conservativity principle in ontology alignment aims at capturing the differences in the ontology classification between the input ontologies and the aligned ontology [36] (i.e., new subsumptions and/or new equivalences among concepts). The conservativity principle despite considering only ontology classification, and not the unrestricted problem addressed by conservative extensions [51], is of high interest because classification is one of the most used features in semantic-enabled applications [25,49].

Conservativity violations may evidence, like consistency violations, erroneous mappings or disagreements in the input ontologies; however, they may also reveal incompleteness in one of the ontologies. Although in the literature there are other approaches that consider these violations as false positives and suggest to fix the input ontologies (see Sect. 8), in this paper we treat them as undesired consequences led by the mappings.

[5] This DL is less expressive than $\mathcal{SROIQ}$, the underlying DL of OWL 2.

The conservativity principle (general notion) states that the aligned ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ should not induce any change in the concept hierarchies of the input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. That is, the sets $\mathsf{diff}^{\approx}_{\Sigma_1}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ and $\mathsf{diff}^{\approx}_{\Sigma_2}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ must be empty for signatures $\Sigma_1 = Sig(\mathcal{O}_1)$ and $\Sigma_2 = Sig(\mathcal{O}_2)$, respectively.

In this paper, we present two less restrictive variants of this principle: *subsumption* and *equivalence* conservativity violations. The *subsumption* variant of the conservativity principle requires $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ 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 addition, we also define violations between concepts that may have been already involved in a subsumption relationship (i.e., resulting in an equivalence between them), denoted as *equivalence conservativity principle violations* or simply *equivalence violations*. Both variants are formally introduced in Definition 2.6.

Definition 2.6 Let $\mathcal{O}_i$ be one of the input ontologies and $\Sigma = Sig(\mathcal{O}_i)\setminus\{\bot, \top\}$ be its signature, let $\mathcal{M}$ be a *coherent* alignment between $\mathcal{O}_1$ and $\mathcal{O}_2$, $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ be the integrated ontology,[6] and let A, B be concepts in Σ. We define two sets of violations of $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ w.r.t. $\mathcal{O}_i$:

- *subsumption violations*, denoted as $\mathsf{subViol}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$, as the set of $A \sqsubseteq B$ axioms satisfying: (i) $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$, (ii) $\mathcal{O}_i \not\models B \sqsubseteq A$, and (iii) there is no C in Σ such that $\mathcal{O}_i \models C \sqsubseteq A$, and $\mathcal{O}_i \models C \sqsubseteq B$;
- *equivalence violations*, denoted as $\mathsf{eqViol}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$, as the set of $A \equiv B$ axioms satisfying: (i) $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} \models A \equiv B$, (ii) $A \sqsubseteq B \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ and/or $B \sqsubseteq A \in \mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$.

Thus, in our setting, an alignment $\mathcal{M}$ violates the conservativity principle if the sets $\mathsf{subViol}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ or $\mathsf{eqViol}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ are not empty. Example 2.1 provides an example of mappings leading to *subsumption* and *equivalence* conservativity violations.

Example 2.1 Consider the alignment $M = \{m_1, m_2, m_3, m_4\}$ provided in Fig. 1. We distinguish the following cases depending on the semantic relation of the mappings:

(i) *Both types of violation.* If both m_1 and m_2 are equivalence mappings (i.e., $\equiv$), then m_1 and m_2 lead to two *subsumption* violations and one *equivalence* violation since they will make $Joint_2$ and $Set\_of\_Joints_2$ to become equivalent, which did not hold any relationship in the original ontology.

(ii) *Only subsumption violation.* In the case, m_1 is an equivalence mapping (i.e., $\equiv$), but m_2 is a subsumption mapping (i.e., $Joint\_Structure_1 \sqsubseteq Set\_of\_Joints_2$), m_1 and m_2 only lead to a *subsumption* violation (i.e., $Joint_2 \sqsubseteq Set\_of\_Joints_2$).

(iii) *Only equivalence violation.* If both m_3 and m_4 are equivalence mappings (i.e., $\equiv$), then these mappings lead to the equivalence of $Anatomical\_Structure_1$ and $Musculoskeletal\_System\_Struct_1$, which represents an *equivalence* violation; however, it does not represent a *subsumption* violation, according to condition (iii) in Definition 2.6, since $Joint\_Structure_1$ is a common descendant of the concepts involved in the novel subsumptions.

(iv) *No violations.* If m_3 is a equivalence mapping (i.e., $\equiv$) but m_4 is a subsumption mapping (i.e., $\sqsubseteq$), then m_3 and m_4 will only lead to the novel subsumption $Musculoskeletal\_System\_Struct_1 \sqsubseteq Anatomical\_Structure_1$, which does not represent a *subsumption* violation for the same reason as described in the previous case.

[6] We assume that $\mathsf{diff}^{\approx}_{\Sigma}(\mathcal{O}_i, \mathcal{O}^{\emptyset}_{\mathcal{O}_1,\mathcal{O}_2}) = \emptyset$.

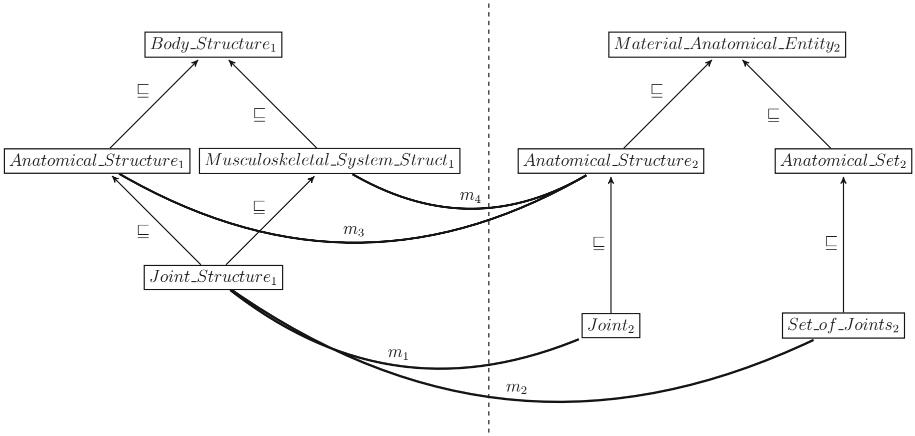

Fig. 1 Example of mappings leading to *subsumption* and *equivalence* conservativity violations. The ontology in the left-hand side represents a fragment of the *SNOMED CT* [69] ontology, while the one in the right-hand side represents a fragment of *FMA* [65] ontology

The (potential) complexity of detecting and solving conservativity violations requires to exploit the intrinsic characteristics of the above two flavours that will result in the development of different methods for their repair. For example, the notion of subsumption violations relies on the assumption of disjointness [67] and it can be reduced to a consistency principle problem (see Sect. 5). Equivalence violations, however, rely on the notion of (unsafe) cycle in the ontology graph and require a different treatment based on graph theory (Sect. 4).

Alignment repair An alignment $\mathcal{M}$ that violates the consistency and/or the conservativity principles can be fixed by removing correspondences from $\mathcal{M}$.[7] This process is referred to as *mapping repair* (or repair for short) and is formally introduced in Definitions 2.7 and 2.8.

Definition 2.7 Let $\mathcal{M}$ be an incoherent alignment (i.e., violates the consistency principle) w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$. A set of mappings $\mathcal{R} \subseteq \mathcal{M}$ is a *mapping repair for the consistency violations* for $\mathcal{M}$ w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$ iff $\mathcal{M}\backslash\mathcal{R}$ is coherent w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$, that is, $\mathcal{O}_{\mathcal{O}_1,\mathcal{O}_2}^{\mathcal{M}\backslash\mathcal{R}} \not\models A \sqsubseteq \bot$, for all $A \in Sig(\mathcal{O}_1) \cup Sig(\mathcal{O}_2)$.

The definition of violation of the conservativity principle, as in Definition 2.6, assumes that the alignment $\mathcal{M}$ does not violate the consistency principle w.r.t. the input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. The main reason for requiring as input a coherent alignment is that unsatisfiable concepts would be subsumed by any other concept, thus leading to a very large number of (misleading) violations of the conservativity principle.

Definition 2.8 Let $\mathcal{M}$ be a coherent alignment that violates the conservativity principle w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$. A set of mappings $\mathcal{R} \subseteq \mathcal{M}$ is a *mapping repair for the conservativity violations* for $\mathcal{M}$ w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$ iff for all $i \in \{1, 2\}$ $\mathsf{subViol}(\mathcal{O}_i, \mathcal{O}_{\mathcal{O}_1,\mathcal{O}_2}^{\mathcal{M}\backslash\mathcal{R}})$ and $\mathsf{eqViol}(\mathcal{O}_i, \mathcal{O}_{\mathcal{O}_1,\mathcal{O}_2}^{\mathcal{M}\backslash\mathcal{R}})$ are empty.

The repair of an alignment usually has several alternatives. Nevertheless, the objective is to remove as few mappings as possible. The notion of minimal repairs, originally introduced by

[7] Note that in this paper, we only target the mappings in the repair process and we consider the input ontologies as immutable. Other approaches like Pesquita et al. [62] question the automatic generation of repairs and suggest to update the ontologies, when necessary, to avoid violations.

Reiter [63] under the name of *diagnoses*, has been introduced to the field of ontology debugging in [68]. In the ontology debugging literature, they are typically referred to as *mapping diagnoses* [54]. A repair or diagnosis for an alignment can be computed by extracting the justifications for the unsatisfiable concepts (e.g., [29,42,67,80]) and selecting a hitting set of mappings to be removed, following a minimality criteria (e.g., the number of removed mappings, or their combined confidence). However, justification-based techniques do not scale when the number of unsatisfiabilities is large (a typical scenario in mapping repair problems [34,76,77]). To address this scalability issue, mapping repair systems usually compute an *approximated repair* using incomplete reasoning techniques (e.g., [33,54,66]). An approximated repair $\mathcal{R}^{\approx}$ does not guarantee that $\mathcal{M}\backslash\mathcal{R}^{\approx}$ does not lead to violations of the consistency and conservativity principle, but it will (in general) significantly reduce the number of violations caused by the original set of mappings $\mathcal{M}$.

2.4 Graph theory notation

This section provides the necessary preliminaries about graph theory, given that graphs will be used to provide a convenient representation of ontologies for reasoning about the detection and repair of conservativity violations.

Graph notions A *directed graph* (digraph) G is a set of vertices V together with a relation A on V.[8] A *weighted digraph* G is a digraph where each arc has a third component, called *weight* of the arc, assuming values in an appropriate structure (e.g., $\mathbb{N}$, $\mathbb{R}$). Given an arc $a = (u, v, c)$, we denote its weight as $w(a) = c$. The function w, when applied to a set of arcs, returns the sum of the weights of the single arcs. An *arc-labelled digraph* $G = (V, A, L)$ is a digraph with an additional component L called the set of labels. Given an arc $a = (u, v, l)$, l is called the label of a, where $l \in L$.

If not differently stated, in the remainder we always refer to *weighted digraphs*, and the placeholder "_" might replace any of the components of an arc, representing an unspecified legal value that the considered component can assume.

Given a digraph $G = (V, A)$, a *subgraph* $G' = (V', A')$ of G is a digraph such that $V' \subseteq V$, and $A' \subseteq A$, and for all arc $a = (u, v, \_) \in A'$, we have that $u, v \in V'$. A *subgraph* of G is said to be *induced* by V' and denoted as $G|_{V'}$, if, for all pair of vertices $u, v \in V'$, $(u, v, \_) \in A'$ iff $(u, v, \_) \in A$.

Paths and cycles We now formally introduce the notion of path and cycles that will be used to model subsumption and equivalence between concepts.

Given a digraph G, a (directed) *path* $\pi = [v_1, \ldots, v_n]$ of G, with $n > 1$, is a sequence of vertices where each pair of vertices v_i and v_{i+1}, with $i \in 1 \ldots n - 1$, is connected by an arc $(v_i, v_{i+1}, \_) \in A$. The *length* of such a path π is $n - 1$. Given two vertices $u, v \in V$, v is *reachable* from u iff a path π of length $m - 1$ exists such that, for some $m > 2$, $v_1 = u$ and $v_m = v$.

Given a digraph G and a directed path $\pi = [v_1, \ldots, v_n]$ of G, we define π as a directed cyclic path (*cycle* in what follows) iff the first and last vertices coincide, i.e., $v_1 = v_n$. We say that a cycle κ is *broken* by the removal of any of its arcs (resp. vertices). We also say that a set of arcs (resp. vertices) breaks a set of cycles iff any of the cycles contains at least one element of the set. Note that we do not consider self-arcs, and therefore, cycles with length equal to 1 are not allowed. Given a digraph G and two cycles κ_1 and κ_2 of G, we define κ_1 and κ_2 as *distinct cycles* if they are not a cyclic permutation one of the other.

[8] In our setting, A is required to be antireflexive as we disallow self-arcs.

Using the notion of cycle, we introduce the notion of *directed acyclic graph* (DAG).

Given an arc $a = (u, v, \_)$, we refer to $\mathcal{V}(a) = \{u, v\}$ as the set containing its source and target vertices. $\mathcal{V}$ naturally extends to sets (resp. sequences) of arcs, as the union of its application on each element of the set (resp. sequence).

Given a path $\pi = [v_1 \dots v_n]$, we refer to $\mathcal{A}(\pi) = \{(v_i, v_{i+1}, \_) \mid i \in [1 \dots n-1]\}$ as the set containing all of the arcs of π. Similarly, given a set of vertices V' and a digraph $G = (V, A)$ such that $V' \subseteq V$, we refer to $\mathcal{A}(V') = \{(u, v, \_) \in A \mid u, v \in V'\}$ as the set containing all the arcs of G between vertices of V'.

Graph Connectivity. Given that paths and cycles are used to encode subsumption and equivalence relations between concepts, graph connectivity represents a natural option for reasoning about transitivity of such relations. The notion of *strongly connected component* (SCC), for instance, splits the graph into equivalence concepts, thus identifying sets of equivalent concepts.

More formally, given a digraph $G = (V, A)$, a SCC of G is a maximal set of vertices $C \subseteq V$ such that for all $u, v \in C$, both u is reachable from v and vice versa. Notice that at least a cycle containing all the elements of the SCC exists, so cycle detection in a digraph G can be reduced to the identification of the SCCs of G. The set of SCCs of a digraph $G = (V, A)$ is denoted as $SCC(G)$. If $SCC(G) = \{V\}$, then G is a *strongly connected digraph*.

Tarjan's Algorithm [81] (Tarjan) finds the SCCs of a digraph $G = (V, A)$ using a single depth-first search with a time complexity in $\mathcal{O}(|V| + |A|)$.

Given a digraph $G = (V, A)$, the *Feedback Edge Set* (*FES*) problem aims at selecting a subset of A, called *feedback (edge) set*, with minimum cardinality, whose removal makes G acyclic. This problem is known to be *NP*-hard [16], as well as its weighted variant, *Weighted Feedback Edge Set* (*WFES*). *WFES* differs from *FES* for what concerns the minimization objective for the feedback set that is required to be minimal w.r.t. the sum of the weights of its elements. *FES* is also equivalent to the *Feedback Vertex Set* (*FVS*) problem [16], where a subset of V that makes G acyclic is sought.

3 Motivating example

In this section, we show the problems caused by the violation of the conservativity principle when integrating ontologies via mappings in a real-world scenario. To this end, we consider, as motivating example, a use case based on the *Optique* project application domain [22, 44].[9] Optique aims at facilitating scalable end-user access to big data in the oil and gas industry. The project is focused around two demanding use cases provided by *Siemens* [45] and *Statoil* [44]. Optique advocates for an Ontology-Based Data Access (OBDA) approach so that end users formulate queries using the vocabulary of a domain ontology instead of composing queries directly against the database. Ontology-based queries (e.g., SPARQL queries) are then automatically rewritten to SQL and executed over the database (e.g., [64]). Ontology entities are linked to the database through *ontology-to-schema* mappings. Ontology-to-schema mappings are, however, out of the scope of the present study, and we therefore only focus on ontology-to-ontology mappings or simply mappings (see Sect. 2.2).

Table 1 shows two simplified fragments of ontologies that are currently being used in the context of Optique. The ontology $\mathcal{O}_1$ has been directly bootstrapped from a relational database (e.g., [38]), and it is linked to the data through (direct) ontology-to-schema mappings. The

[9] http://www.optique-project.eu.

Table 1 Simplified fragments of two ontologies in the oil and gas domain

	Ontology $\mathcal{O}_1$		Ontology $\mathcal{O}_2$
α_1	$WellBore \sqsubseteq \exists belongsTo.Well$	β_1	$Exploration\_well \sqsubseteq Well$
α_2	$WellBore \sqsubseteq \exists hasOperator.Operator$	β_2	$Explor\_borehole \sqsubseteq Borehole$
α_3	$WellBore \sqsubseteq \exists locatedIn.Field$	β_3	$Appraisal\_exp\_borehole \sqsubseteq Explor\_borehole$
α_4	$AppraisalWellBore \sqsubseteq WellBore$	β_4	$Appraisal\_well \sqsubseteq Well$
α_5	$ExplorationWellBore \sqsubseteq WellBore$	β_5	$Field \sqsubseteq \exists hasFieldOperator.Field\_operator$
α_6	$Operator \sqsubseteq Owner$	β_6	$Field\_operator \sqcap Owner \sqsubseteq Field\_owner$
α_7	$Operator \sqsubseteq Company$	β_7	$Company \sqsubseteq Field\_operator$
α_8	$Field \sqsubseteq \exists hasOperator.Company$	β_8	$Field\_owner \sqsubseteq Owner$
α_9	$Field \sqsubseteq \exists hasOwner.Owner$	β_9	$Borehole \sqsubseteq Continuant \sqcup Occurrent$

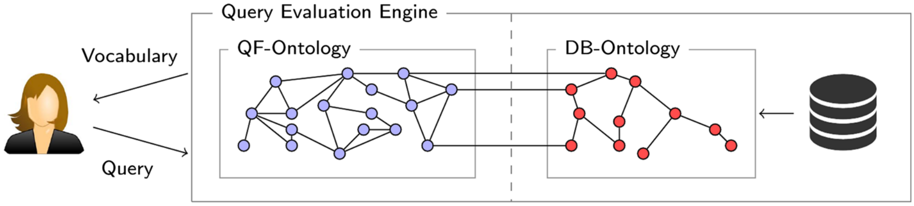

Fig. 2 Ontology alignment in an OBDA scenario

ontology $\mathcal{O}_2$, instead, is a domain ontology based on the Norwegian Petroleum Directorate (NPD) FactPages[10] [72], preferred by the Optique end users to feed the Optique's visual query formulation interface [79].[11]

The integration via ontology alignment of $\mathcal{O}_1$ and $\mathcal{O}_2$ is required since the vocabulary in $\mathcal{O}_2$ is used to formulate queries (i.e., QF-Ontology), but only the vocabulary of $\mathcal{O}_1$ is connected to the database (i.e., DB-Ontology), as depicted in Fig. 2. Consider the set of mappings $\mathcal{M}$ in Table 2 between $\mathcal{O}_1$ and $\mathcal{O}_2$ generated by an off-the-shelf ontology alignment system. Mappings are represented as 4-tuples (see Definition 2.1); 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.

The integrated ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$, however, violates the conservativity principle (see Table 3). According to Definition 2.6, entailments σ_1-σ_3, σ_6 and σ_7 represent subsumption violations, while entailments σ_4 and σ_5 are equivalence violations. Note that σ_4 and σ_5 do not belong to the set of subsumption 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. $\mathcal{O}_2$).

In Fig. 3, the portion of the graph representation of $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ involved in conservativity violations is shown. Dashed arcs represent inferred axioms, while bold arcs are those involved in equivalence violations. Each non-inferred arc is labelled with its confidence value.

Example 3.1 provides an instance of query over the vocabulary of $\mathcal{O}_2$.

[10] http://factpages.npd.no/factpages/.

[11] Optique uses OWL 2 QL ontologies for query rewriting, while the query formulation may be based on much richer OWL 2 ontologies. The axioms that fall outside the OWL 2 QL profile are either approximated or not considered for the rewriting.

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

	e_1	e_2	n	ρ
Mappings $\mathcal{M}$				
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 violations

σ	Entailment:	follows from:	subViol?	eqViol?
σ_1	$\mathcal{O}_2{:}Explor\_borehole \sqsubseteq \mathcal{O}_2{:}Exploration\_well$	m_3, m_4	Yes	No
σ_2	$\mathcal{O}_1{:}AppraisalWellBore \sqsubseteq \mathcal{O}_1{:}ExplorationWellBore$	β_3, m_4, m_5	Yes	No
σ_3	$\mathcal{O}_2{:}Field\_operator \sqsubseteq \mathcal{O}_2{:}Field\_owner$	$\alpha_6, \beta_6, m_7, m_{10}$	Yes	No
σ_4	$\mathcal{O}_1{:}Company \equiv \mathcal{O}_1{:}Operator$	$\alpha_7, \beta_7, m_7, m_8$	No	Yes
σ_5	$\mathcal{O}_2{:}Field\_operator \equiv \mathcal{O}_2{:}Company$			
σ_6	$\mathcal{O}_1{:}Company \sqsubseteq \mathcal{O}_1{:}Owner$	σ_4, α_6	Yes	No
σ_7	$\mathcal{O}_2{:}Company \sqsubseteq \mathcal{O}_2{:}Field\_owner$	σ_3, σ_5	Yes	No

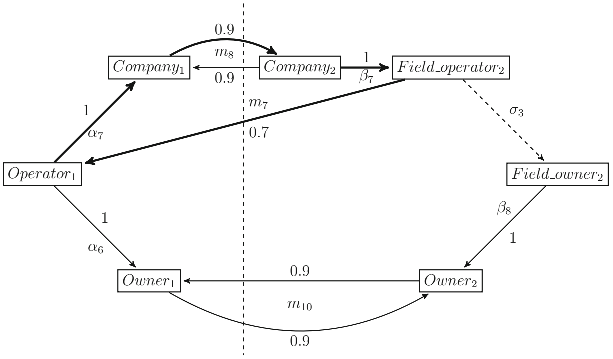

Fig. 3 Graph representation of the fragment of the aligned ontology involved in conservativity violations

Example 3.1 Consider the following simple conjunctive query expressed in datalog notation, $CQ(x) \leftarrow \mathcal{O}_2{:}Well(x)$. The query asks for wells and has been formulated using the vocabulary of $\mathcal{O}_2$. The query is rewritten, according to the ontology axioms and mappings $\beta_1, \beta_4, m_1, m_3, m_4$ in $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$, into the following union of conjunctive queries:

$$UCQ(x) \leftarrow \mathcal{O}_2{:}Well(x) \cup \mathcal{O}_1{:}Well(x) \cup \mathcal{O}_2{:}Exploration\_well(x) \cup \\ \mathcal{O}_2{:}Appraisal\_well(x) \cup \mathcal{O}_1{:}ExplorationWellBore(x) \cup \\ \mathcal{O}_2{:}Explor\_borehole(x)$$

Since only the vocabulary of $\mathcal{O}_1$ is linked to the data, the union of conjunctive queries could be simplified as $UCQ(x) \leftarrow Well(x) \cup ExplorationWellBore(x)$, which will clearly lead to non desired results. The original query was only asking for wells, while the rewritten query will also return data about exploration wellbores.

Example 3.1 shows that the quality of the mappings in terms of conservativity violations may directly affect the quality of the query results in an OBDA context. Therefore, the detection and repair of these violations arise as an important quality assessment step in Optique. Conservativity violations, however, may represent *false positives*, that is, entailments that bring new and valid knowledge. In these cases, the detection and (suggested) repair of conservativity violations can be seen as an input of a subsequent manual revision where the bootstrapped and the domain ontologies may be updated with novel subsumption and/or equivalence axioms. Note that changes in the bootstrapped ontology will necessarily imply a revision of the ontology-to-schema mappings. Conservativity violations, however, must always be avoided in the case the bootstrapped and the domain ontologies are considered as immutable by Optique end users.

4 Detecting and repairing equivalence violations

This section introduces the relevant properties of our aligned ontology graph representation as well as the formal definition and analysis of the problem of computing a minimal diagnosis to repair equivalence violations. Finally, we present our repair algorithm EqRepair which relies on logic programming.

As stated in Definitions 2.6 and 2.8, we expect as input an alignment $\mathcal{M}$ that is coherent w.r.t. the input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$.

4.1 Aligned ontology and graph representation

Definition 4.1 formalizes a variant of the aligned ontology given in Definition 2.3. This variant enables a more efficient detection phase for equivalence violations. Differently from Definition 2.3, here we perform a classification step on the input ontologies before computing the aligned ontology, in order to obtain a graph representation that allows the detection of equivalence violations by simply analysing the graph, without further use of an OWL 2 reasoner.

Definition 4.1 Given two (input) ontologies, namely $\mathcal{O}_1$, $\mathcal{O}_2$, and an alignment $\mathcal{M}$ between them, consider an ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ such that $Sig(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}) = Sig(\mathcal{O}_1) \cup Sig(\mathcal{O}_2)$, and $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} = \mathcal{C}l(\mathcal{O}_1) \cup \mathcal{C}l(\mathcal{O}_2) \cup \mathcal{M}$, where $\mathcal{C}l$ represents the classification of the given ontology as introduced in Sect. 2.1. We define $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ as the *aligned ontology* w.r.t. $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathcal{M}$, or simply the aligned ontology, when no confusion arises.

Note that we do not compute the closure of $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ since this will prevent our repair algorithm to isolate the axioms (i.e., mappings) causing an equivalence violation.

An example of aligned ontology is shown in Example 4.1.

Table 4 Ontologies and alignment for Example 4.1

$\mathcal{C}l(\mathcal{O}_1)$	$\mathcal{C}l(\mathcal{O}_2)$	$\mathcal{M}$
$Laptop \sqsubseteq PC$		
$Ultrabook \sqsubseteq Laptop \sqcap LightObject$	$Ultrabook \sqsubseteq Notebook$	$\langle Laptop_1, Notebook_2, \sqsupseteq, 0.7\rangle$
$Ultrabook \sqsubseteq Laptop^*$	$Notebook \sqsubseteq Computer$	$\langle Ultrabook_1, Ultrabook_2, \equiv, 1\rangle$
$Ultrabook \sqsubseteq LightObject^*$	$Ultrabook \sqsubseteq Computer^*$	$\langle PC_1, Computer_2, \sqsubseteq, 0.9\rangle$
$Ultrabook \sqsubseteq PC^*$		

Inferred axioms are marked with an asterisk

Example 4.1 Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two input ontologies and $\mathcal{M}$ be an alignment between them. Table 4 shows the classification of the input ontologies (inferred axioms are marked with an asterisk) and the mappings. The aligned ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$ is therefore equal to $\{Laptop_1 \sqsubseteq PC_1$, $Ultrabook_1 \sqsubseteq Laptop_1 \sqcap LightObject_1$, $Ultrabook_1 \sqsubseteq Laptop_1$, $Ultrabook_1 \sqsubseteq LightObject_1$, $Ultrabook_1 \sqsubseteq PC_1$, $Ultrabook_2 \sqsubseteq Notebook_2$, $Notebook_2 \sqsubseteq Computer_2$, $Ultrabook_2 \sqsubseteq Computer_2$, $Notebook_2 \sqsubseteq Laptop_1$, $Ultrabook_1 \equiv Ultrabook_2$, $PC_1 \sqsubseteq Computer_2\}$. Note that subscripts are only used to emphasize the “provenance” of named concepts.

Note that our repair algorithm does not directly compute a diagnosis using the aligned ontology, and it rather works on a graph representation[12] of this ontology, presented in Definition 4.2.

Definition 4.2 Given an ontology $\mathcal{O}$, its *graph representation*, denoted as $G(\mathcal{O}) = (V, A)$, is a digraph characterized by the following properties:

- each concept $C \in N_C(\mathcal{O})$ has an associated vertex $v_C \in V$,
- each axiom of the form $D \sqsubseteq E \in \mathcal{O}$ (resp. $D \sqsupseteq E$), with $D, E \in N_C(\mathcal{O})$ distinct concepts, has an associated arc (v_D, v_E, c) (resp. (v_E, v_D, c)) in A,
- each axiom of the form $D \equiv E \in \mathcal{O}$, with $D, E \in N_C(\mathcal{O})$ distinct concepts, has an associated pair of arcs $\{(v_D, v_E, c), (v_E, v_D, c)\}$ in A.

The third component of each arc (denoted as c above) is always equal to 1 if $\mathcal{O}$ is not an aligned ontology; otherwise, it is equal to 1 for the axioms of the input ontologies, while it is equal to the confidence of the considered mapping for the corresponding arcs.

When no confusion arises, we interchangeably refer to an ontological concept and its associated vertex and to an ontology and its associated graph representation. Note that subsumption/equivalence axioms between a concept and itself are not allowed in the graph representation, because the set of arcs is defined using an antireflexive relation. Similarly, we will interchangeably refer to axioms/mappings and their corresponding elements in the graph representation.

An example of graph representation, associated with the aligned ontology of Example 4.1, is shown in Fig. 4.

4.2 Paths and cycles

Given the semantics of the arcs in the graph representation, a path from a vertex u to a vertex v implies that $\mathcal{O}$ entails that the concept associated with u is subsumed by the concept

[12] Although several proposals for graph formalisms for representing DL ontologies exist in the literature (e.g., [58]), we provided a simplified variant specifically tailored to capture equivalence violations.

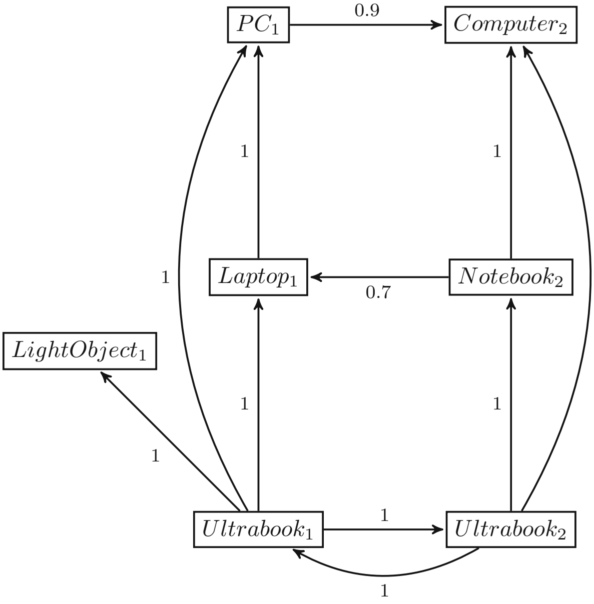

Fig. 4 Graph representation for the aligned ontology of Example 4.1

associated with v, as expressed by Proposition 4.1. Analogously, Proposition 4.2 states that a cycle in the graph representation implies that $\mathcal{O}$ entails that all the concepts associated with the vertices of the cycle are equivalent. Note that the other direction of the following propositions holds only if the graph representation is built on top of an ontology closed w.r.t. classification, but this is not the case for our technique. Therefore, we do not require that the graph representation of the aligned ontology reflects all its subsumption entailments (equivalently, it suffices that testing subsumption on the graph representation is sound, even if not complete).

Proposition 4.1 *Let $\mathcal{O}$ be an ontology and let $G(\mathcal{O}) = (V, A)$ be its graph representation. If a path $\pi = [v_{A_1}, \ldots, v_{A_n}]$ with $n > 1$ exists in $G(\mathcal{O})$, then $\mathcal{O} \models A_1 \sqsubseteq A_n$, with A_1, A_n distinct concepts.*

Proof The proof directly follows from the semantics of the arcs in the graph representation and by transitivity of the subsumption relation. □

Proposition 4.2 *Let $\mathcal{O}$ be an ontology and let $G(\mathcal{O}) = (V, A)$ be its graph representation. If a cycle $\kappa = [v_{A_1}, \ldots, v_{A_n}, v_{A_1}]$ with $n \geq 1$ exists in $G(\mathcal{O})$, then $\mathcal{O} \models A_i \equiv A_j$, with $i, j \in [1 \ldots n]$ and A_i, A_j distinct concepts.*

Proof Given a pair of vertices v_{A_i}, v_{A_j}, we define two paths $\pi_{ij} = [v_{A_i}, \ldots, v_{A_j}]$ and $\pi_{ji} = [v_{A_j}, \ldots, v_{A_i}]$ such that vertices v_{A_p}, v_{A_q} are consecutive in π_{ij}, π_{ji}, only if the same holds in κ, with $i, j, p, q \in [1 \ldots n]$. Given that all the π_{ij} and π_{ji} are paths, we can apply Proposition 4.1. □

Safe cycles In Definition 4.3, safe cycles are introduced, that is, cycles of the aligned ontology that do not violate the conservativity principle. This notion is key to discriminate between cycles representing or not an equivalence violation, and it is also at the basis of the encoding of equivalence violations in terms of our graph representation. Intuitively, a cycle is safe if the projection on each of the two input ontologies is either a single vertex, or a cycle traversing all the vertices of such projection exists. A special case of safe cycles is represented by cycles already existing in the input ontologies. The different kinds of safe cycles stem from the combination of the following conditions on the two projections of the cycle over the input ontologies: a projection is empty (condition (i)), a projection consists of a single vertex (condition (ii)), a cycle traversing all the vertices of a projection exists (condition (iii)). For instance, a (safe) cycle is local to the input ontology $\mathcal{O}_i$ if its projection on $\mathcal{O}_i$ satisfies condition (iii), while the other projection is empty (condition (i) holds). This implies an equivalence entailment (involving all the concepts of the cycle) already in the input ontology $\mathcal{O}_i$ that clearly prevents the existence of an equivalence violation among such concepts.

Definition 4.3 Let $\kappa = [u_1, \ldots, u_n, u_1]$ be one of the cycles of a graph representation $G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$, with $n > 1$. We define κ as a *safe cycle* iff for some $i \in \{1, 2\}$ we have that: either (i) $|\mathcal{V}(\kappa) \cap V_{\mathcal{O}_i}| = 0$, or (ii) $|\mathcal{V}(\kappa) \cap V_{\mathcal{O}_i}| = 1$, or (iii) a cycle κ'_i in $G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})|_{V_{\mathcal{O}_i}}$ exists, such that $\mathcal{V}(\kappa) \cap V_{\mathcal{O}_i} \subseteq \mathcal{V}(\kappa'_i)$. We further differentiate between *local* ((i) and (iii) hold), *trivial* (only (ii) holds), *partially trivial* ((ii) and (iii) hold) and *nontrivial safe cycles* (only (iii) holds). We generically refer to safe cycles, except local safe cycles, as *global safe cycles*. A cycle that is not safe is defined as an *unsafe cycle*.

In Example 4.2, an example is provided for each different kinds of cycles.

Example 4.2 Consider the graph representation of Fig. 5. We first present the (different kind of) safe cycles. $[Food_2, Nourishment_2, Cooking_2, Food_2]$ is a local safe cycle, $[Food_1, Food_2, Nourishment_2, Food_1]$ is partially trivial, and $[Food_1, Food_2, Food_1]$ is trivial. Instead, $[Food_2, Nourishment_2, PetFood_1, Food_1, Food_2]$ is an unsafe cycle, but adding the axiom $Food_1 \sqsubseteq PetFood_1$ to $\mathcal{O}_1$ would turn it into a nontrivial safe cycle.

Starting from the results of Proposition 4.2, we can characterize a restricted version of the conservativity principle using graph-theoretical concepts only, applied on the graph representation, without the need to refer to the aligned ontology. Despite the aforementioned approximations, our repair technique for equivalence violations is effective in practice, as experimentally verified in Sect. 7.2.

Theorem 4.1 *Let $\mathcal{O}_1, \mathcal{O}_2$ be two (input) ontologies, and let $\mathcal{M}$ be an alignment between them. Let also $G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}) = (V, A)$, $G(\mathcal{O}_1) = (V_{\mathcal{O}_1}, A_{\mathcal{O}_1})$ and $G(\mathcal{O}_2) = (V_{\mathcal{O}_2}, A_{\mathcal{O}_2})$ be the graph representations associated with $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$, $\mathcal{O}_1$ and $\mathcal{O}_2$, respectively. If there exists an unsafe cycle κ^u in $G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ and the vertices associated with concepts C, D (namely V_C, V_D) belong to κ^u, then a violation of the conservativity principle w.r.t. equivalence, involving concepts C and D, exists in $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$.*

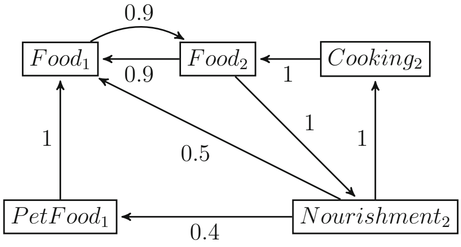

Fig. 5 A graph representation including both safe and unsafe cycles

Proof For all the $v_C, v_D \in c$, if we have an (unsafe) cycle κ^u in $G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$, then $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} \models C \equiv D$ holds. For all $i \in \{1, 2\}$, if an unsafe cycle κ^u exists, then no cycle κ' exists in $G(\mathcal{O}_i)$ such that $\mathcal{V}(\kappa^u) \cap V_{\mathcal{O}_i} \subseteq \mathcal{V}(\kappa')$. This requires that at least one vertex $v_E \in \kappa^u$ exists such that it does not belong to κ', which corresponds to having that, for all $i \in \{1, 2\}$ and for some $E, F \in N_C(\mathcal{O}_i)$, $\mathcal{O}_i \not\models E \equiv F$ and $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} \models E \equiv F$. This represents an equivalence violation, thus proving the soundness of the detected violations using the graph representation. □

Proposition 4.2 relates cycles in the graph representation to equivalence axioms in the aligned ontology. We have only soundness guarantee, given that without classifying the aligned ontology equivalences that are not reflected in the graph representation may exist.

Theorem 4.1 guarantees that a method detecting and correcting *all* the unsafe cycles on the graph representation is sound w.r.t. violations of the conservativity principle w.r.t. equivalence in the aligned ontology. This is proved by verifying that detected unsafe cycles effectively encode equivalence violations.

We remark also that, once the graph representation is built, we do not need the support of a standard reasoner. This feature is at the basis of the scalability of our approach.

4.3 Diagnoses for equivalence conservativity violations

The goal of our repair algorithm is, starting from an aligned ontology, to detect violations to the conservativity principle w.r.t. equivalence and to compute a diagnosis. In order to obey to the well-known *principle of minimal change* , the diagnosis is required to have a minimal weight, w.r.t. a metric capturing the amount of lost information. In our context, a standard minimality criterion for defining diagnosis weight (the quantity to minimize) is the sum of the weights of the arcs associated with the (removed) mappings. In Definition 4.4, we formalize a diagnosis as the set of arcs of the graph representation of an aligned ontology that, once removed, breaks all the unsafe cycles.

Definition 4.4 Let $\mathcal{M}$ be an alignment such that $G(\mathcal{O}^{\mathcal{M}})$ has unsafe cycles $\{\kappa^u_1, \ldots, \kappa^u_n\}$. Let also $\Delta \subseteq \mathcal{M}$ be an alignment. Δ is a *diagnosis* for $\mathcal{M}$ iff for all $i \in [1 \ldots n]$, $\Delta \cap \mathcal{A}(\kappa^u_i) \neq \emptyset$, that is, the graph representation $G(\mathcal{O}^{\mathcal{M} \setminus \Delta})$ has no unsafe cycles.

Alternatively, we refer to a diagnosis for $\mathcal{M}$ as the diagnosis for the aligned ontology $\mathcal{O}^{\mathcal{M}}$ or its graph representation $G(\mathcal{O}^{\mathcal{M}})$. In Definition 4.5, we formalize diagnosis minimality (required by the principle of minimal change), where given a diagnosis $\Delta = \{a_0, \ldots, a_n\}$, its weight, denoted as $w(\Delta)$, is equal to $\sum_{i=1}^{n} w(a_i)$.

Definition 4.5 Let Δ be a diagnosis for an alignment $\mathcal{M}$. Δ is a *minimal diagnosis* for $\mathcal{M}$ iff there is no diagnosis Δ' for $\mathcal{M}$ such that $w(\Delta') < w(\Delta)$.

Before introducing the notion of problematic SCC in Definition 4.6, key for the definition of an efficient detection method for unsafe cycles, we need to define the *projection of a SCC $S \in SCC(G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}))$ on an input ontology $\mathcal{O}_i$*, denoted as $\Pi_{\mathcal{O}_i}(S)$, as $S \cap V_{\mathcal{O}_i}$, where $G_{\mathcal{O}_i} = (V_{\mathcal{O}_i}, A_{\mathcal{O}_i})$ is the graph representation of $\mathcal{O}_i$. We denote with $SCC_{local}(G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}))$ the set of SCCs of $G(\mathcal{O}^{\emptyset}_{\mathcal{O}_1,\mathcal{O}_2})$.

Definition 4.6 A SCC $S \in SCC(G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}))$ is *problematic* iff $\Pi_{\mathcal{O}_1}(S) \notin SCC_{local}(G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})) \vee \Pi_{\mathcal{O}_2}(S) \notin SCC_{local}(G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}))$. The set of *problematic SCCs* of $G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2})$ is denoted as $pSCC(G(\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}))$.

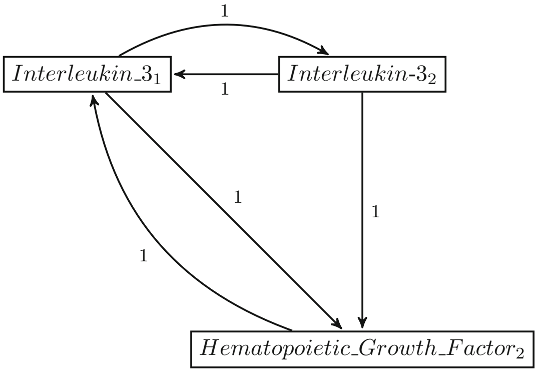

Fig. 6 Example of cycle defined by the *UMLS* 2012 alignment between *FMA* and *NCI* ontologies. The equivalence of the three vertices is entailed by the aligned ontology

Example 4.3 Consider the graph represented in Fig. 6. It is a subgraph of the aligned ontology $\mathcal{O}^{\mathcal{UMLS}}$ representing the *UMLS* 2012 alignment, namely $\mathcal{UMLS}$, between ontologies $\mathcal{O}_1$, $\mathcal{O}_2$, corresponding to the *FMA* [65] and *NCI* [24]. The graph represents a problematic SCC because its projection on $\mathcal{O}_2$ does not belong to the local SCCs of $\mathcal{O}_2$ since no cycle between its two vertices exists. It cannot traverse another SCC, because otherwise the two SCCs would be a single one.

Example 4.4 Consider two input ontologies $\mathcal{O}_1 = \{MolarTooth_1 \sqsubseteq Tooth_1\}$ and $\mathcal{O}_2 = \{Tooth19_2 \sqsubseteq MolarTooth_2\}$ and their obvious associated signatures. Consider also the alignment $\{\langle MolarTooth_1, MolarTooth_2, \equiv, 1\rangle, \langle Tooth_1, Tooth19_2, \equiv, 0.7\rangle\}$. The resulting digraph contains a SCC including all the four vertices, but none of the two projections on the input ontologies are SCCs. As a consequence, all these vertices are equivalent in the aligned ontology. From the domain knowledge, however, we know that $Tooth_1 \sqsubseteq MolarTooth_1$ and $Tooth_2 \sqsubseteq Tooth19_2$ do not hold.

The set of *problematic mappings* is the set of mappings between vertices of a problematic SCC, the only arcs that can appear in diagnoses.

The notion of problematic SCCs enables an alternative diagnosis definition. Using the notion of problematic SCC, Definitions 4.7 and 4.8 propose an alternative definition for diagnosis, w.r.t. that given in Definition 4.4.

Definition 4.7 Let Δ be an alignment such that $\Delta \subseteq \mathcal{M}$, and let $G(\mathcal{O}^{\mathcal{M}\setminus\Delta}) = (V, A')$ be a graph representation such that $A' = A\setminus\Delta$. Δ is a *diagnosis* for $\mathcal{M}$ w.r.t. $G(\mathcal{O}^{\mathcal{M}})$ iff $pSCC(G(\mathcal{O}^{\mathcal{M}\setminus\Delta})) = \emptyset$.

Definition 4.8 Let Δ be a diagnosis for $\mathcal{M}$, let $S \in pSCC(G(\mathcal{O}^{\mathcal{M}}))$ be a problematic SCC, and let Δ' be a nonempty subset of Δ. Δ' is a *diagnosis for a SCC S* iff for all S_i in $SCC(G(\mathcal{O}^{\mathcal{M}\setminus\Delta'}))$ such that $S_i \subseteq S$, we have that $S_i \notin pSCC(G(\mathcal{O}^{\mathcal{M}\setminus\Delta'}))$.

Clearly, the reformulation does not affect any of the properties of diagnosis illustrated in this section. Example 4.5 shows a minimal diagnosis for a problematic SCC.

Example 4.5 Consider the scenario of Example 4.3 and the subgraph of Fig. 6, representing a problematic SCC. A minimal diagnosis for the problematic SCC is $\{\langle Interleukin\_3_1, Hematopoietic\_Growth\_Factor_2, 1\rangle\}$.

4.4 Problem statement for diagnosis computation

This section introduces the *MAP-WFES* problems as an extension of the well-known *WFES* problem, which removes unsafe cycles only for solving the equivalence violations, while preserving local cycles. The computational complexity of the *MAP-WFES* problem is analysed in "Diagnosis computation complexity" in section "Appendix".

MAP-WFESDefinition. Let $G = (V, A)$ be a digraph such that $V = V_1 \cup V_2$, with $V_1 \cap V_2 = \emptyset$. We denote $\mathcal{M}$, the set of *mappings*, the subset of A whose arcs are of the form $(u, v, \_)$, such that for all $i, j \in \{1, 2\}$ and $i \neq j$ we have that $u \in V_i$, $v \in V_j$.

MAP-WFES aims at computing, given as input the digraph G, a minimum weight feedback edge set $\Delta \subseteq \mathcal{M}$ such that no unsafe cycles exist in $G' = (V, A \backslash \Delta)$.

Proposition 4.3 relates diagnosis computation to the *MAP-WFES* problem introduced in this section. Specifically, it states that computing a diagnosis for a graph representation reduces to solving *MAP-WFES* on it.

Proposition 4.3 *Computing a minimal diagnosis for an aligned ontology $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$, that is, solving all the violations to the conservativity principle w.r.t. equivalence, can be reduced to solving an instance of MAP-WFES.*

Proof From Theorem 4.1 we know that all the violations of the conservativity principle w.r.t. equivalence in the aligned ontology result in unsafe cycles in its graph representation. Therefore breaking all unsafe cycles is equivalent to computing a diagnosis. From the definition of *MAP-WFES*, we have that no unsafe cycles can exist in the graph resulting from the application of the computed diagnosis. Given that, with the exception of computing a minimal diagnosis, *MAP-WFES* has no other constraints, it computes such a diagnosis, and this concludes the proof. □

Since *MAP-WFES* is a *NP*-hard problem (as shown in "Diagnosis computation complexity" in section "Appendix"), and given the average size of an aligned ontology, computing a diagnosis would be, in practice, intractable. For this reason, our approach decomposes the problem into independent subproblems (i.e., computing a local diagnosis for each problematic SCC), following a *divide et impera* strategy. A (minimal) global diagnosis is then computed from the (minimal) local diagnoses of the single problematic SCCs. The optimality and correctness of this decomposition are investigated in "Decomposability of equivalence violations diagnosis computation" in section "Appendix".

4.5 Solving equivalence violations using logic programming

This section introduces the method for computing minimal diagnoses using logic programming and a detailed description of the EqRepair algorithm.

Minimal Diagnosis Using ASP. We now describe how a minimal diagnosis can be computed using Answer Set Programming (*ASP*) programs.[13, 14] The facts input to the *ASP* problems are of the following kinds: (i) Vertices are represented using a binary predicate $vtx(X, O)$, where X is a string representing the vertex label and $O \in \{1, 2\}$ encodes the index of the input ontology the represented concept belongs to, (ii) arcs are represented using a quaternary

[13] We use the syntax of *Lparse 1.0*, a parser for logic programs used as a front-end by different logic programming solvers, more details at http://www.tcs.hut.fi/Software/smodels/.

[14] Although alternative frameworks could have been employed (e.g., Constraint Logic Programming), we have adopted *ASP* as it is known to be well suited for graph-related problems and to produce compact and easy-to-understand solutions [10].

Listing 1 *ASP* facts encoding the problematic SCC of Fig. 3.

```
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).
```

Listing 2 *ASP* program computing a minimal diagnosis for the *MAP-WFES* problem.

```
r0: #domain vtx(X,O). #domain vtx(Y,P). #domain vtx(Z,Q).
r1: reaches(X,Y) :– edge(X,Y,C,M), not_removed(edge(X,Y,C,M)), X!=Y.
r2: reaches(X,Z) :– reaches(X,Y), edge(Y,Z,C,M), not_removed(edge(Y,Z,C,M)), X!=Y, Y!=Z, X!=Z.
r3: reachesSafe(X,Y) :– edge(X,Y,C,0), O=P, X!=Y.
r4: reachesSafe(X,Z) :– reachesSafe(X,Y), edge(Y,Z,C,0), O=P, X!=Y, Y!=Z, X!=Z.
r5: not_removed(edge(X,Y,C,M)) | removed(edge(X,Y,C,M)) :– edge(X,Y,C,M), X!=Y .
r6: not_removed(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 ] .
r10: #hide . #show removed/1.
```

predicate $edge(X, Y, C, M)$, where X, Y are vertices (that is, for some $O, P \in \{1, 2\}$, $vtx(X, O)$ and $vtx(Y, P)$ hold), C is an integer encoding the arc weight in a range $[0 \dots 100]$, while M is a Boolean flag for differentiating arcs that correspond to axioms ($M = 0$) from those corresponding to mappings ($M = 1$).

When we refer to the execution of an *ASP* program on a graph representation (resp. an aligned ontology), we always implicitly refer to its execution on a set of facts encoding the graph representation (resp. aligned ontology).

Example 4.6 In the following, we provide an example of encoding of a graph into a set of facts that can be used in conjunction with the *ASP* programs for solving the *MAP-WFES* problems. To this end, we rely on our motivating example presented in Sect. 3. The encoding of the problematic SCC of Fig. 3 corresponds to the set of facts presented in Listing 1. The execution of the *ASP* program of Listing 2, in conjunction with the aforementioned facts, is equal to the mapping m_7 from Table 2 (i.e., the candidate mapping(s) to be removed to solve the violation(s)).

A minimal diagnosis can be computed using the *ASP* program shown in Listing 2.

Rule r0 "types" variables X, Y or Z as vertex id (into the vtx predicate they appear). Rules r1 states that if an unremoved arc $(X, Y, \_)$ exists, then vertex X $reaches$ vertex Y, rule r2 states that $reaches$ is a transitive predicate. Similarly, rule r3 states that if an arc $(X, Y, \_)$ exists, that is neither a mapping nor removed, then vertex X $reachesSafe$ vertex Y, and rule r4 makes $reachesSafe$ a transitive predicate.

In order to obtain a valid solution, each arc has to be either removed or not removed (rule r5), and only mappings can be removed (rule r6). Additionally, rule r8 forbids the existence of unsafe cycles, identified by rule r7 (if a vertex unsafely reaches itself, then we have at least an unsafe cycle). Among the valid solutions, rule r9 minimizes diagnosis weight, and the output model is restricted to $removed$ predicate only by means of rule r10 that corresponds to the computed diagnosis.

EqRepair Algorithm. The EqRepair algorithm[15] (Algorithm 1) takes as input two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, and a (coherent) alignment $\mathcal{M}$ between them. The graph representation G of the

[15] In [73], this algorithm is referred to as CycleBreaker.

Algorithm 1 EqRepair Algorithm for *equivalence* conservativity violations

```
Input: $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; $\mathcal{M}$: input (coherent) mappings
Output: $\Delta$: diagnosis
1: $G \leftarrow$ createDigraph($\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}$)
2: $SCCs_i \leftarrow$ tarjan($G, \mathcal{O}_i$)
3: $globalSCCs \leftarrow$ tarjan($G$)
4: $mappings \leftarrow \emptyset$                                  ▷ Map of SCCs' mappings
5: $\Delta \leftarrow \emptyset$                                     ▷ Diagnosis for $\mathcal{M}$
6: for each $S$ in $globalSCCs$ do
7:     if $\neg \bigwedge_{i=1}^{2} \Pi_{O_i}(S) \in SCCs_i$ then
8:         $mappings(S) \leftarrow$ extractMappings($S$)
9:         $\Delta \leftarrow \Delta \cup$ solver($S$)
10:    end if
11: end for
12: return $\Delta$
```

aligned ontology w.r.t. $\mathcal{O}_1, \mathcal{O}_2$ and $\mathcal{M}$, is built by means of createDigraph function (line 1 of Algorithm 1). The function builds a digraph G representing the aligned ontology associated with the input ontologies $\mathcal{O}_1$, $\mathcal{O}_2$ and alignment $\mathcal{M}$. In accordance with Definition 4.2, the vertices of this graph are the named concepts of the two ontologies, and its arcs are the axioms/mappings involving them.[16]

After the creation of the digraph, the SCCs of the input ontologies (line 2) and that of the aligned ontology (line 3) are computed by means of the *Tarjan's* algorithm (introduced in Sect. 2.4). The algorithm then detects which SCCs of the aligned ontology are problematic. By Definition 4.6, it suffices to test whether at least one of the two projections on the input ontologies of each considered SCC is not a local SCC (line 7).

The minimal diagnosis for the current SCC, namely S, is computed by the solver function that runs an *ASP* solver over a program representing an instance of the *MAP-WFES* problem associated with S (line 9 of Algorithm 1). The diagnosis for S is then obtained, using a translation, from the solution of the aforementioned problem. Theorem 4.1 and Proposition 4.3 detail, respectively, the correctness of the detection and the solution computation, as described above.

The global diagnosis results from the union of the local diagnoses, that is, the diagnoses of the single SCCs (the correctness proof is given in Proposition 10.5, "Decomposability of equivalence violations diagnosis computation" in section "Appendix").

5 Detecting and repairing Subsumption violations

Our technique for coping with subsumption violations is based on the reduction of the conservativity principle repair problem to a consistency principle repair problem (i.e., a mapping incoherence repair problem) through the assumption of disjointness [67]. Currently, our method reuses and adapts the structural indexing and reasoning techniques implemented in LogMap (an ontology matching and mapping repair system [33,37,39]). However, alternative mapping repair systems could be used, such as ALCOMO [54] or AML [17]. Note that, to the best of our knowledge, these mapping repair systems have only focused on solving violations of the consistency principle.

Algorithm 2 shows *SubRepair* algorithm, the proposed method for detecting and correcting subsumption violations. SubRepair algorithm expects as input an alignment $\mathcal{M}$ that is

[16] For sake of space, the algorithm, which directly follows from Definition 4.2, is omitted. The interested reader can find it in [73], Algorithm 12.

Algorithm 2 SubRepair algorithm for *subsumption* conservativity violations

```
Input: $\mathcal{O}_1$, $\mathcal{O}_2$: input ontologies; $\mathcal{M}$: input (coherent) mappings
Output: $\mathcal{R}^{\approx}$: approximate repair; $disj$: number of disjointness rules
1: $\langle \mathcal{P}_1, \mathcal{P}_2 \rangle \leftarrow$ propositionalEncoding($\mathcal{O}_1, \mathcal{O}_2$)
2: $SI_1 \leftarrow$ structuralIndex($\mathcal{O}_1$)
3: $SI_2 \leftarrow$ structuralIndex($\mathcal{O}_2$)
4: $\mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2} \leftarrow \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$    ▷ The aligned ontology is computed
5: $SI_{\mathcal{U}} \leftarrow$ structuralIndex($\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$)
6: $\langle \mathcal{P}^d_1, disj_1 \rangle \leftarrow$ disjointAxiomsExtension($\mathcal{P}_1, SI_1, SI_{\mathcal{U}}, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$)    ▷ See Algorithm 3
7: $\langle \mathcal{P}^d_2, disj_2 \rangle \leftarrow$ disjointAxiomsExtension($\mathcal{P}_2, SI_2, SI_{\mathcal{U}}, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1,\mathcal{O}_2}$)
8: $\langle \mathcal{M}', \mathcal{R}^{\approx} \rangle \leftarrow$ mappingRepair($\mathcal{P}^d_1, \mathcal{P}^d_2, \mathcal{M}$)    ▷ See Algorithm 2 in [39]
9: $disj \leftarrow disj_1 + disj_2$
10: return $\langle \mathcal{R}^{\approx}, disj \rangle$
```

coherent w.r.t. the input ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, according to Definitions 2.6 and 2.8. SubRepair algorithm outputs the number of added disjointness during the process $disj$ and an (approximate) repair $\mathcal{R}^{\approx}$. The following paragraphs describe the techniques used at each step.

Propositional Horn Encoding. The ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ are encoded as the Horn propositional formulas, $\mathcal{P}_1$ and $\mathcal{P}_2$ (line 1 in Algorithm 2). For example, the concept hierarchy provided by an OWL 2 reasoner (e.g., [23,43]) is encoded as $A \rightarrow B$ rules, while the explicit disjointness relationships between concepts are represented as $A_i \wedge A_j \rightarrow \bot$. Note that the input mappings $\mathcal{M}$ can already be seen as propositional implications. This encoding is key to the mapping repair process.

Example 5.1 Consider the ontologies and mappings in Tables 1 and 2. The axiom β_6 is encoded as $Field\_operator \wedge Owner \rightarrow Field\_owner$, 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 indexing Given that queries over the structural relationships of ontologies are heavily employed in our approach, we rely on the optimized structural index of LogMap [33, 39], based on the interval labelling schema techniques presented in [1].

Specifically, the structural index exploits an optimized data structure for storing directed acyclic graphs (DAGs), and it allows us to answer many entailment queries over the concept hierarchy as an index lookup operation and hence without the need of an OWL 2 reasoner (after the initial classification of the ontology). This kind of index has demonstrated to significantly reduce the cost of answering taxonomic queries [7,59] and disjointness relationships queries [33,37]. A formal definition of the structural index is provided in [73], Definition 6.4, Section 6.3.2.

Hence, the concept hierarchies provided by an OWL 2 reasoner and the explicit disjointness axioms of $\mathcal{O}_1$ and $\mathcal{O}_2$ are efficiently indexed into the structural index (lines 2 and 3 in Algorithm 2).

Disjointness addition In order to reduce the 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 ontology $\mathcal{O}_i$. However, the insertion of additional disjointness axioms δ may lead to unsatisfiable concepts in $\mathcal{O}_i \cup \{\delta\}$, as shown in Example 5.2.

Example 5.2 Consider the axiom β_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.

Algorithm 3 disjointAxiomsExtension function for disjointness axioms extension

```
Input: $\mathcal{P}$: propositional theory; $SI$: structural index; $SI_{\mathcal{U}}$: structural index of the aligned ontology
Output: $\mathcal{P}^d$: extended propositional theory; $disj$: number of disjointness rules
1: $disj \leftarrow 0$
2: $\mathcal{P}^d \leftarrow \mathcal{P}$
3: for each $A \rightarrow B$ in subsumptionViolations($SI, SI_{\mathcal{U}}$) do        ▷ As in Definition 2.6
4:     if not ($SI$.areDisj($A, B$)) then
5:         $\mathcal{P}^d \leftarrow \mathcal{P}^d \cup \{A \wedge B \rightarrow$ false$\}$
6:         $SI \leftarrow SI$.updateIndex($A \sqcap B \rightarrow \bot$)
7:         $disj \leftarrow disj + 1$
8:     end if
9: end for
10: return $\langle \mathcal{P}^d, disj \rangle$
```

In order to detect whether each candidate disjointness axiom leads to an unsatisfiability, a non-naive algorithm requires to make an extensive use of an OWL 2 reasoner to check whether there are new unsatisfiable concepts in $\mathcal{O}_i \cup \{\delta\}$. In large ontologies, however, this approach can be prohibitive.

Our method, instead, exploits the propositional encoding and structural indexing of the input ontologies. Thus, checking whether $\mathcal{O}_i \cup \{\delta\}$ contains unsatisfiable concepts is restricted to the Horn propositional case. The safety conditions for disjointness clauses in addition to a (satisfiable) Horn propositional formula are given in Proposition 5.1, while the proof can be found in [73], Proposition 6.2, Section 6.3.2. These conditions can easily be tested with the structural index.

Proposition 5.1 *Given a satisfiable Horn propositional formula $\mathcal{P}$, the addition of a (j-th) disjointness clause $A \wedge B \rightarrow \bot$ will not cause any (potential) unsatisfiability of the propositions in $\mathcal{P}$ iff: (i) neither $\mathcal{P} \models A \rightarrow B$ nor $\mathcal{P} \models B \rightarrow A$ holds, (ii) no proposition C exists such that $\mathcal{P} \models C \rightarrow A$ and $\mathcal{P} \models C \rightarrow B$.*

Algorithm 2 extends the propositional formulas $\mathcal{P}_1$ and $\mathcal{P}_2$ with disjointness rules of the form $A \wedge B \rightarrow \bot$ (lines 6–7). The disjointness addition follows Proposition 5.1 and guarantees that, for every proposition A in the extended propositional formula $\mathcal{P}_i^d$ (with $i \in \{1, 2\}$), the formula $\mathcal{P}_i^d \cup \{\top \rightarrow A\}$ is satisfiable. This does not necessarily hold if the disjointness axioms are added to the OWL 2 ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, as discussed above.

Note that the addition of all possible disjointness rules may be prohibitive for large ontologies and unnecessary (see [75]). Thus, one should only add disjointness where a subsumption violation occurs, i.e., adding a disjointness axiom between each pair of concepts $A, B \in \mathcal{O}_i$ (with $i \in \{1, 2\}$) such that $A \sqsubseteq B \in \mathsf{subViol}(\mathcal{O}_i, \mathcal{O}^{\mathcal{M}}_{\mathcal{O}_1, \mathcal{O}_2})$, as in Definition 2.6. Algorithm 3 implements this idea for the Horn propositional case and extensively exploits the structural index to identify the subsumption violations (line 3 of Algorithm 3). This algorithm requires as input the structural index of the integrated ontology and thus its classification with an OWL 2 reasoner (line 5 in Algorithm 2). The classification time of the integrated ontology is known to be typically much higher than that of the input ontologies individually [34]. However, this was not a bottleneck in our experiments, as shown in Sect. 7.2.

Mapping repair The step 8 of Algorithm 2 uses the mapping (incoherence) repair algorithm of LogMap, for the extended Horn propositional formulas $\mathcal{P}_1^d$ and $\mathcal{P}_2^d$, and the input mappings $\mathcal{M}$. The mapping repair process exploits the Dowling–Gallier (D&G) algorithm [11,21] for propositional Horn satisfiability (refer to [73], Section 6.3, for more details) and checks, for every proposition A of a given formula $\mathcal{P}$, the satisfiability of the propositional formula $\mathcal{P}_A = \mathcal{P} \cup \{\top \rightarrow A\}$. Satisfiability of $\mathcal{P}_A$ is checked in worst-case linear time in the size of $\mathcal{P}_A$, and the number of D&G calls is also linear in the number of propositions in $\mathcal{P}$. In case of unsatisfiability, the variant of the algorithm implemented in LogMap also allows us to record *conflicting* mappings involved in the unsatisfiability, which will be considered for the subsequent repair process.[17]

The repair is performed by removing some of the identified conflicting mappings. To this aim, the algorithm selects an approximation of the minimal hitting set (w.r.t. total mapping confidence) between the set of conflicting mappings, for the different unsatisfiabilities that are detected. In case of multiple options, the mapping confidence will be used as a differentiating factor.[18] Example 5.3 provides an example of mapping repair for solving a subsumption violation.

Example 5.3 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 3. For example, $\mathcal{P}_2^d$ includes the rule $\psi = \mathcal{O}_2{:}Well \wedge \mathcal{O}_2{:}Borehole \rightarrow \mathsf{false}$. The mapping repair algorithm identifies the propositional theory $\mathcal{P}_1^d \cup \mathcal{P}_2^d \cup \mathcal{M} \cup \{\mathsf{true} \rightarrow \mathcal{O}_1{:}ExplorationWellbore\}$ as unsatisfiable. This is due to the combination of the mappings m_3 and m_4, the propositional projection of axioms β_1 and β_2, and the rule ψ. 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).

6 Properties of the repair methods and combined approach

We have presented two methods for the detection and correction of violations of the conservativity principle. The repair of subsumption conservativity violations (which are reduced to a consistency violations) is based on the Dowling–Gallier algorithm for propositional Horn satisfiability, whereas equivalence conservativity violations are addressed using a combination of graph theory and logic programming.

Soundness and (In)completeness. Both methods are sound (the violations that are detected are indeed violations if considering the full expressiveness of the input ontologies), but incomplete, since the used approximate projections of the input ontologies (i.e., Horn propositional and graph encodings) may lead to some violations being missed.

Algorithm 2 computes a repair $\mathcal{R}^{\approx}$ such that $\mathcal{M}' = \mathcal{M} \backslash \mathcal{R}^{\approx}$ is coherent with respect to $\mathcal{P}_1^d$ and $\mathcal{P}_2^d$ (according to the propositional case of Definition 2.5). Furthermore, the propositional theory $\mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{M}'$ does not contain any subsumption violation with respect to $\mathcal{P}_1$ and $\mathcal{P}_2$ (according to the propositional case of Definition 2.6). However, our encoding is incomplete, and we cannot guarantee that $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}'$ does not contain any subsumption conservativity violations w.r.t. $\mathcal{O}_1$ and $\mathcal{O}_2$.

Analogously, Algorithm 1 computes a diagnosis (i.e., minimal mapping repair) Δ such that $\mathcal{M}' = \mathcal{M} \backslash \Delta$ does not lead to equivalence violations with respect to the graph encoding of the aligned ontology presented in Definition 4.1. However, the considered encoding is incomplete since only the classification of the input ontologies in isolation is considered.

[17] Note that, as for the case of EqRepair, we do not compute the classification of $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ since this will prevent our extension of D&G to identify and record the mappings involved in an unsatisfiability (i.e., a subsumption violation).

[18] In scenarios where the confidence of the mapping is missing (e.g., in reference or manually created mapping sets) or unreliable, our mapping repair technique computes fresh confidence values based on the locality principle [36].

Algorithm 4 ConsRepair, a multi-strategy approach to repair conservativity violations

```
Input: $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; $\mathcal{M}$: input (coherent) mappings; $rs$: repair strategy;
Output: $\mathcal{R}^{\approx}$: approximate repair; $disj$: number of disjointness rules
1: $disj \leftarrow 0$
2: switch $rs$ do
3:     case equivalence repair
4:         $\mathcal{R}^{\approx} \leftarrow \mathsf{EqRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$        ▷ Algorithm 1
5:     case subsumption repair
6:         $\langle \mathcal{R}^{\approx}, disj \rangle \leftarrow \mathsf{SubRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$        ▷ Algorithm 2
7:     case equivalence and subsumption repair
8:         $\mathcal{R}_1^{\approx} \leftarrow \mathsf{EqRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$
9:         $\langle \mathcal{R}_2^{\approx}, disj \rangle \leftarrow \mathsf{SubRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M} \backslash \mathcal{R}_1^{\approx})$
10:        $\mathcal{R}^{\approx} \leftarrow \mathcal{R}_1^{\approx} \cup \mathcal{R}_2^{\approx}$
11:    case subsumption and equivalence repair
12:        $\langle \mathcal{R}_1^{\approx}, disj \rangle \leftarrow \mathsf{SubRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$
13:        $\mathcal{R}_2^{\approx} \leftarrow \mathsf{EqRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M} \backslash \mathcal{R}_1^{\approx})$
14:        $\mathcal{R}^{\approx} \leftarrow \mathcal{R}_1^{\approx} \cup \mathcal{R}_2^{\approx}$
15: return $\langle \mathcal{R}^{\approx}, disj \rangle$
```

Thus, as for the subsumption violations, we cannot guarantee that $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}'$ does not contain any equivalence violation.

Nevertheless, incompleteness is mitigated thanks to the classification of the input ontologies using full reasoning (see Definition 4.1 and the propositional encoding in Sect. 5). Furthermore, our evaluation suggests that the number of remaining violations after repair is typically small (see Sect. 7.2).

Combined Algorithm. Despite the differences between subsumption and equivalence violations, a mutual influence between them exists. An example is given in Table 3 of Sect. 3, where subsumption violations are caused by the existence of equivalence violations in the same aligned ontology. Nonetheless, subsumption violations can also be responsible for the existence of equivalence violations.

It is evident, from the aforementioned example, that the application of a repair step targeting a particular violation kind can solve as a side effect violations of other kinds.

In Algorithm 4, we provide a multi-strategy algorithm to repair conservativity violations. The algorithm allows combining the approaches targeting subsumption and equivalence violations. We expect that the combination of the repair strategies will impact the performance in terms of required time and number of solved violations. Furthermore, we also expect the order of the strategies will influence the results.

Algorithm 4, as SubRepair and EqRepair algorithms, expects as input two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, and a coherent alignment $\mathcal{M}$. Additionally, it also expects the required repair strategy rs (e.g., subsumption repair first and then equivalence repair). As output, the algorithm provides the (approximate) repair $\mathcal{R}^{\approx}$ and the number of added disjointness $disj$ in the SubRepair algorithm. When SubRepair is not executed, $disj = 0$. In the case of using a combined strategy, the repaired mappings by the SubRepair (resp. EqRepair) algorithm will be given as input of the EqRepair (resp. SubRepair) algorithm; see lines 12 and 13 (resp. 8 and 9) in Algorithm 4.

7 Experimental evaluation

In this section, we evaluate the feasibility of using our combined approach to correct conservativity violations in practice.[19,20] To this end, we have conducted the evaluation in Algorithm 5 over the ontologies and mapping sets (i.e., reference alignments and alignments computed by participating systems) of the 2012–2014 campaigns of the Ontology Alignment Evaluation Initiative (*OAEI*):

(i) *Anatomy* the Anatomy dataset involves the Adult Mouse Anatomy (*MO*) ontology and a fragment of the *NCI* ontology describing human anatomy. The reference alignment has been manually curated [86]. From 2012 to 2014, we have gathered 47 different alignments computed by participating systems.

(ii) *Conference* this dataset uses a collection of 16 ontologies from the domain of academic conferences [83]. Currently, there are 21 manually created mapping sets among 7 of the ontologies. From 2012 to 2014, we have gathered 1, 104 different alignments computed by participating systems.

(iii) *LargeBio* this dataset includes the biomedical ontologies *FMA*, *NCI* and (a fragment of) *SNOMED CT*, and three reference mapping sets based on the *UMLS* [5]. From 2012 to 2014, we have gathered 122 alignments computed by participating systems.

(iv) *Library* this OAEI dataset includes the real-world thesauri STW and TheSoz from the social sciences. The reference mappings have been manually validated. From 2012 to 2014, we have gathered 32 alignments computed by participating systems.

We have run the evaluation algorithm for (i) each of the OAEI tasks (i.e., pair of ontologies) described above and (ii) each of the reference and computed alignments available in each OAEI task, which resulted in 1, 331 executions and 5, 324 calls (1, 331 x 4 repair strategies) to the repair method (line 9 in Algorithm 5).

In the conducted evaluation, we have used the OWL 2 reasoner HermiT [23] to classify the input and aligned ontologies. In a few cases, HermiT failed in classifying the aligned ontology and we used the OWL 2 EL reasoner ELK [43] in order to provide an approximation of the classification. The use of this approximation had a minor impact in the overall results.

Section 7.1 briefly comments on the steps followed by Algorithm 5. In Sect. 7.2, we present and discuss the results of our repair methods in terms of computation times, computed repairs and corrected violations. Note that we have grouped the results according to the origin of the alignments: reference and computed by participating systems. Section 7.3 analyses and compares the impact of the conservativity repair strategies on the alignment quality in terms of precision, recall, and f-measure.

7.1 Evaluation algorithm steps

Algorithm 5 expects as input two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, a (not necessarily coherent) alignment $\mathcal{M}$, and the set of supported repair strategies RS (as in Algorithm 4). Note that the evaluation algorithm includes two pre-processing steps of the input mappings and ontologies (lines 1 and 2 of Algorithm 5).

[19] The complete source code of the proposed algorithms and the performed experiments is available at https://github.com/asolimando/logmap-conservativity/.

[20] The test environment consisted of a desktop computer equipped with 32*GB DDR* 3 *RAM* at 1333*MHz*, and an *AMD Fusion FX 4350* (quad-core, each running at 4.2*GHz*) as *CPU*. The dataset is stored on a 128*GB SSD*, where the operating system *Ubuntu* (12.04, 64-bit version) is also installed. Our prototype can run with less than 8GB for the majority of the considered tests; we, however, allocate 26GB of RAM for the JVM in order to minimize the influence of the garbage collector on the recorded temporal measurements.

Algorithm 5 Conducted evaluation

```
Input: $\mathcal{O}_1$, $\mathcal{O}_2$: input ontologies; $\mathcal{M}$: alignment between $\mathcal{O}_1$ and $\mathcal{O}_2$; RS: repair strategies;
    Pre-processing steps:
1: $\langle \mathcal{O}_1, \mathcal{O}_2 \rangle \leftarrow \mathsf{ModuleExtraction}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$
2: $\mathcal{M} \leftarrow \mathsf{ConsistencyRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M})$
    Create aligned ontology:
3: $\mathcal{O}^{\mathcal{M}} \leftarrow \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$    ▷ Definition 2.3
    Keep problem size:
4: Store $|\mathsf{Sig}(\mathcal{O}_1)|$, $|\mathsf{Sig}(\mathcal{O}_2)|$, and $|\mathcal{M}|$
    Compute number of initial violations:
5: $\mathsf{subViol}_i \leftarrow |\mathsf{subViol}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}})| + |\mathsf{subViol}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}})|$    ▷ Subsumption violations, Definition 2.6
6: $\mathsf{eqViol}_i \leftarrow |\mathsf{eqViol}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}})| + |\mathsf{eqViol}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}})|$    ▷ Equivalence violations, Definition 2.6
7: $\mathsf{diff}_i^{\approx} \leftarrow |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_1)}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}})| + |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_2)}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}})|$    ▷ General notion, Definition 2.4
8: for each rs in RS do    ▷ Repair strategies as in Algorithm 4
        Compute repair (Algorithm 4):
9:      $\langle \mathcal{R}^{\approx}, disj \rangle \leftarrow \mathsf{ConsRepair}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}, rs)$    ▷ Keep times to compute repair $t_r$ and disjointness rules $t_d$
        Create new (repaired) aligned ontology:
10:     $\mathcal{M}' \leftarrow \mathcal{M} \backslash \mathcal{R}^{\approx}$
11:     $\mathcal{O}^{\mathcal{M}'} \leftarrow \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}'$
        Compute number of remaining violations:
12:     $\mathsf{subViol}_r \leftarrow |\mathsf{subViol}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}'})| + |\mathsf{subViol}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}'})|$
13:     $\mathsf{eqViol}_r \leftarrow |\mathsf{eqViol}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}'})| + |\mathsf{eqViol}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}'})|$
14:     $\mathsf{diff}_r^{\approx} \leftarrow |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_1)}(\mathcal{O}_1, \mathcal{O}^{\mathcal{M}'})| + |\mathsf{diff}^{\approx}_{\mathsf{Sig}(\mathcal{O}_2)}(\mathcal{O}_2, \mathcal{O}^{\mathcal{M}'})|$
15: end for
```

Module extraction Modules are a general technique for ensuring scalability of ontology-based algorithms. The main intuition is that a (logic-based) module for a given ontology $\mathcal{O}$ w.r.t. a seed signature Σ is a subset of $\mathcal{O}$ that preserves the entailment relation over axioms expressed using Σ. In our approach, in order to reduce the size of the problem, we (optionally) extract two modules [8,26], one for each input ontology, using the entities involved in the input mappings $\mathcal{M}$ as seed signatures. When no confusion arises, we directly refer to these modules as the input ontologies.

Consistency repair As already mentioned, our (combined) approach expects to work on coherent alignments. In order to meet this requirement, we perform a preliminary consistency repair as a two-step process. First, the consistency repair facility of LogMap [39] is used to repair the aligned ontology. Then, any unsolved incoherence is detected through an OWL 2 reasoner and solved by computing a single justification [28] and removing the mapping with least confidence appearing in it. This last step is iteratively applied until no more incoherences are present in the aligned ontology. This consistency repair process is sound and complete, but it does not guarantee the optimal minimization of the total confidence of the removed mappings. However, for the purposes of our evaluation, this two-step process is sufficient.

Problem characteristics Line 4 keeps the size of the problem (i.e., number of entities of the input ontologies and size of the input mapping). The number of initial violations prior repair is also stored: subsumption violations (line 5), equivalence violations (line 6) and conservativity violations following the general notion (line 7), which fully relies on the approximation of the deductive difference (Definition 2.4).

Conservativity repair Algorithm 5 computes a repair for each of the four supported strategies keeping the times to compute the repair (t_r) and to add the disjointness rules (t_d) in the relevant cases (line 9). Once the repair has been computed, it also calculates the violations that have been missed (lines 10-14). Although our approach is incomplete, the number of remaining violations is typically negligible compared to the initial number of violations reported in lines 5-7.

7.2 Evaluation on computed and reference alignments

The result of applying the evaluation algorithm to the ontologies and reference alignments of the OAEI campaign is presented in Table 5.[21] Table 6 reports the same evaluation over the computed alignments by participating systems in the OAEI 2012–2014 campaigns. Tables 5a and 6a show the characteristics of the problem (i.e., average size of input ontologies and mappings, and number of violations prior repair). The last column (#$\mathcal{M}$) represents the number of matching tasks in each of the tracks. For example, there are 122 different alignments in the *largebio* track computed by participating systems with an average size of $11,789$ mappings per alignment. The results compiling the output of several repairs (i.e., over several alignments) are presented in the tables with the format *mean (standard deviation)*. For example, for the *conference* track (Table 5a), since #$\mathcal{M}$=21 (i.e., there are 21 alignments), we report the average size of (initial) subsumption violations (2.19) and the standard deviation of their size (3.5).

Next, we first analyse the results of the two basic repair strategies in isolation (i.e., EqRepair and SubRepair), and then, we compare with the results of the combined approaches (i.e., EqRepair first and then SubRepair, and vice versa).

Equivalence The experimental results considering EqRepair algorithm can be summarized as follows:

(i) The sum of the detection and repair time of EqRepair is very low due to the linear cost of the detection technique and the efficient parallelization of the diagnosis computation.
(ii) The computed repairs are typically of limited size (less than 10%), but can reach a significant portion of the the original alignment. For example, in the *library* track (Table 6b), it removes on average 22% of the mappings. This is due to the large number of violations led by the alignments computed by participating systems.
(iii) The equivalence violations are completely removed for *anatomy* and *library* tracks and practically removed for *largebio* and *conference* compared to the initial number of equivalence violations (see Tables 5b and 6b).
(iv) The removal of equivalence violations, as expected, has also an important impact w.r.t. the remaining subsumption violations, specially when dealing with mapping sets computed by participating systems in the OAEI's *library* track (see Table 6b).

This set of experiments confirms the effectiveness and efficiency of the detection and repair algorithms for equivalence violations.

Subsumption The experimental results on the behaviour of SubRepair algorithm can be summarized as follows:

(i) Despite the large number of subsumption violations, the required time t_d to detect violations and add disjointness rules (i.e., cost of Algorithm 3) is very low (see Tables 5c and 6c). The results also show that t_d is directly influenced by the size of the input ontologies, more than by the number of violations.
(ii) Repair times t_r are typically small and they do not represent a bottleneck in spite of the large number of added disjointness rules. An exception is represented by the *library*

[21] Note that the reference mappings of the OAEI campaign are already coherent w.r.t. the test case ontologies, and thus, the consistency repair step was not necessary.

Table 5 Results for *OAEI* reference alignments, grouped by track

OAEI track	Problem size			Initial violations			$\#\mathcal{M}$
	$\|\mathsf{Sig}(\mathcal{O}_1)\|$	$\|\mathsf{Sig}(\mathcal{O}_2)\|$	$\|\mathcal{M}\|$	$\mathsf{subViol}_i$	$\mathsf{diff}_i^{\approx}$	eqViol_i	
(a) Problem characteristics							
Anatomy	2747	3306	3032	1321	1335	26	1
Conference	110 (31)	121 (44)	29 (10)	2.19 (3.5)	2.24 (3.48)	0.1 (0.3)	21
Largebio	21,693 (25,739)	14,674 (8811)	19,813 (15,457)	234,111 (284,869)	242,517 (297,275)	2536 (3096)	3
Library	6575	8376	6322	42,045	42,872	895	1
OAEI track	Solution size	Time	Remaining violations				
	$\|\mathcal{R}^{\approx}\|$	t_r(s)	$\mathsf{subViol}_r$	$\mathsf{diff}_r^{\approx}$	eqViol_r		
(b) Equivalence repair in isolation							
Anatomy	25	0.61	1,259	1,271	0		
Conference	0.1 (0.3)	0.05 (0.05)	2.05 (3.4)	2.05 (3.4)	0		
Largebio	1801 (1,927)	15 (5.28)	164,377 (187,554)	168,819 (194,184)	18 (32)		
Library	627	2.58	30,006	30,408	0		

Table 5 continued

OAEI track	Solution size		Times		Remaining violations		
	#disj	$\lvert\mathcal{R}^{\approx}\rvert$	t_d(s)	t_r(s)	$\mathsf{subViol}_r$	$\mathsf{diff}_r^{\approx}$	eqViol_r
(c) Subsumption repair in isolation							
Anatomy	1321	317	0.68	0.35	0	3	1
Conference	2.19 (3.5)	1.05 (1.47)	0.05 (0.05)	0	0	0.05 (0.22)	0.05 (0.22)
Largebio	234,111 (284,869)	6625 (5749)	28 (22)	164 (243)	170 (213)	688 (944)	123 (175)
Library	42,045	2183	2.73	9.31	0	38	10
(d) Equivalence repair first and then subsumption repair (Eq.→Sub)							
Anatomy	1259	325	0.68	0.53	0	2	0
Conference	2.05 (3.4)	1.05 (1.32)	0.05 (0.05)	0.01 (0.01)	0	0	0
Largebio	164,377 (187,554)	6683 (5912)	27 (20)	114 (162)	36 (57)	121 (68)	0.33 (0.58)
Library	30,006	2140	2.45	8.77	0	6	0
(e) Subsumption repair first and then equivalence repair (Sub.→Eq.)							
Anatomy	1321	318	0.68	0.47	0	2	0
Conference	2.19 (3.5)	1.1 (1.45)	0.05 (0.05)	0.01 (0.01)	0	0	0
Largebio	234,111 (284,869)	6,730 (5897)	28 (22)	165 (244)	48 (51)	158 (141)	1 (1.73)
Library	42,045	2,193	2.73	9.73	0	9	0

Cells compiling the results of several repairs are presented with the format *"mean (std deviation)"*

Table 6 Results for *OAEI* 2012–2014 computed alignments, grouped by track

OAEI track	Problem size			Initial violations			$\#\mathcal{M}$
	$\lvert\mathsf{Sig}(\mathcal{O}_1)\rvert$	$\lvert\mathsf{Sig}(\mathcal{O}_2)\rvert$	$\lvert\mathcal{M}\rvert$	$\mathsf{subViol}_i$	$\mathsf{diff}_i^{\approx}$	eqViol_i	
(a) Problem characteristics							
Anatomy	2747	3306	2,607 (816)	6562 (12,844)	6619 (12,911)	137 (340)	47
Conference	110 (30)	121 (43)	33 (35)	17 (86)	17 (87)	1.27 (7.74)	1104
Largebio	18,340 (19,857)	13,450 (6995)	11,789 (8729)	154,566 (237,174)	159,711 (245,746)	716 (1100)	122
Library	6575	8376	6363 (3168)	5,121,627 (8×10^6)	5,153,135 (8×10^6)	516,289 (10^6)	32
OAEI track	Solution size	Time	Remaining violations				
	$\lvert\mathcal{R}^{\approx}\rvert$	t_r (s)	$\mathsf{subViol}_r$	$\mathsf{diff}_r^{\approx}$	eqViol_r		
(b) Equivalence repair in isolation							
Anatomy	82 (190)	4.24 (14)	4,011 (7888)	4030 (7,913)	0		
Conference	0.85 (5.83)	0.26 (3.48)	11 (42)	11 (43)	0.05 (1.16)		
Largebio	453 (661)	13 (11)	110,323 (153,602)	113,571 (158,770)	18 (49)		
Library	1401 (1866)	24 (37)	90,154 (53,457)	91,996 (54,483)	0		

Table 6 continued

OAEI track	Solution size		Times		Remaining violations		
	#disj	$\lvert\mathcal{R}^{\approx}\rvert$	t_d(s)	t_r(s)	$\mathsf{subViol}_r$	$\mathsf{diff}_r^{\approx}$	eqViol_r
(c) Subsumption repair in isolation							
Anatomy	6562 (12,844)	396 (413)	0.77 (0.4)	1.22 (2.6)	0	2.23 (1.99)	0.47 (1.02)
Conference	17 (86)	3.4 (10)	0.17 (1.45)	0.01 (0.09)	0.15 (1.33)	0.18 (1.34)	0.03 (0.2)
Largebio	154,566 (237,174)	2897 (2847)	25 (18)	134 (286)	79 (151)	234 (348)	35 (61)
Library	5,121,627 (8 × 10^6)	2865 (2196)	48 (75)	1966 (3690)	0	67 (54)	16 (14)
(d) Equivalence repair first and then subsumption repair (Eq.→Sub)							
Anatomy	4011 (7888)	384 (400)	0.76 (0.4)	4.39 (14)	0	1.6 (0.85)	0
Conference	11 (42)	3.43 (10)	0.16 (1.41)	0.22 (3.48)	0.16 (1.24)	0.16 (1.24)	0
Largebio	110,323 (153,602)	2851 (2823)	24 (17)	87 (145)	52 (130)	104 (140)	0.43 (1.45)
Library	90,154 (53,457)	2896 (2329)	39 (62)	42 (42)	0	3.81 (4)	0
(e) Subsumption repair first and then equivalence repair (Sub.→Eq.)							
Anatomy	6562 (12,844)	397 (414)	0.77 (0.4)	1.31 (2.6)	0	1.51 (0.91)	0
Conference	17 (86)	3.43 (10)	0.17 (1.45)	0.01 (0.09)	0.15 (1.33)	0.15 (1.33)	0
Largebio	154,566 (237,174)	2,926 (2890)	25 (18)	135 (287)	53 (129)	107 (144)	0.51(1.48)
Library	5,121,627 (8 × 10^6)	2881 (2204)	48 (75)	1967 (3690)	0	15 (10)	0

Cells compiling the results of several repairs are presented with the format *"mean (std deviation)"* track (Table 6c), where an average repair time of 33 min is required. However, the runtime is reasonable considering the impressive average number of violations (i.e., $5M$).

(iii) The computed repairs $\mathcal{R}^{\approx}$ can be aggressive and the average mapping removal ranges from 3% (*conference* track, Table 5c) up to 45% of the original alignment (*library* track, Table 6c). From Tables 5c and 6c, the expected positive correlation between the repair size and the number of detected violations (i.e., added disjointness rules #disj) clearly emerges.

(iv) Subsumption violations are completely removed in the *anatomy* and *library* cases and almost completely removed in the *conference* and *largebio* tracks, with less than 0.1% (on average) of unsolved violations. As a side effect, equivalence violations are also reduced to a minimum. For example, for one of the computed alignments between *iasted* and *sigkdd* ontologies in the *conference* track, we fail to detect and repair the conservativity violation $iasted{:}Hotel\_fee \sqsubseteq iasted{:}Reg\_fee$. The justifications of this violation reveal that an inverse property axiom and an existential restriction are behind this novel entailment. These OWL 2 constructors, however, fall outside the Horn propositional and graph projections of the input ontologies currently implemented in our techniques.

(v) The number of missed violations is only slightly higher when considering the general notion of the conservativity principle (see $\mathsf{diff}_r^{\approx}$ columns in Tables 5c and 6c), which suggests that our (approximate) variant based on the assumption of disjointness is suitable in practice. Furthermore, the number of unsolved violations using this notion is negligible (<0.25 % on average).

This set of experiments basically confirms the effectiveness and efficiency of the detection and repair algorithms for subsumption violations. This evaluation also shows that the violation detection algorithm is more heavily influenced by the size of the involved ontologies and alignments, while the time required by the repair algorithm is strongly correlated with the number of violations to repair.

Combined The experimental results on the behaviour of the different flavours of our combined approaches can be summarized as follows:

(i) In Tables 5d, e, and 6d, e, we can see that the number of unsolved violations, as expected, is lowered w.r.t. applying one of the repair methods in isolation. In *anatomy* and *conference*, the results are slightly better when applying SubRepair algorithm first, while in *largebio* and *library*, the unsolved violations are smaller when applying EqRepair algorithm first; nevertheless, the differences are negligible.

(ii) The detection and disjointness addition time t_d are comparable in both combined variants, and it is not therefore influenced by the repair application order. However, the number of required disjointness rules #disj is significantly reduced when applying EqRepair algorithm first.

(iii) Total repair times t_r when applying SubRepair first are basically the same as the times when applying SubRepair in isolation. However, the repair times when applying EqRepair first for the *largebio* and *library* tracks are significantly reduced, specially for the case of *library*. This is due to the fact that the SubRepair algorithm, in the presence of SCCs, generates a large number of almost equivalent repair plans that pose a problem to the plan selection method. The solution presented in EqRepair is more efficient in these cases since it focuses on the problematic SCCs.

(iv) The computed repairs $\mathcal{R}^{\approx}$ are of comparable size, on average smaller when equivalence repair is applied first. An exception is, however, represented by the alignments computed by participating systems in the *library* track.

This set of experiments confirms the increased effectiveness, with a small loss in efficiency, of the combined repair approaches, w.r.t. the single repair methods.

7.3 Repair effects on alignment quality

Figures 7, 8, 9 and 10 show the average impact (in terms of precision, recall and f-measure) of applying a conservativity repair over the computed alignments by participating systems in the OAEI 2012–2014 campaigns. The results are grouped by track.

The impact of alignment repair is computed as the percentual of gain (resp. loss for negative values) for each measure computed for a repaired alignment, compared to the same measure computed for the original alignment. The measures are computed w.r.t. the *OAEI* reference alignments. Additionally, we filter any alignment without conservativity violations, because the empty repair always implies a void gain.

Such impact of the conservativity repair is presented in Figures 7, 8, 9 and 10 for each flavour of our repair algorithm, where results over individual alignments are grouped by track. The boxplots presented in such figures are characterized by a "box" (i.e., the range in between the first and third percentile), the "median" for the results, and the "whiskers" which

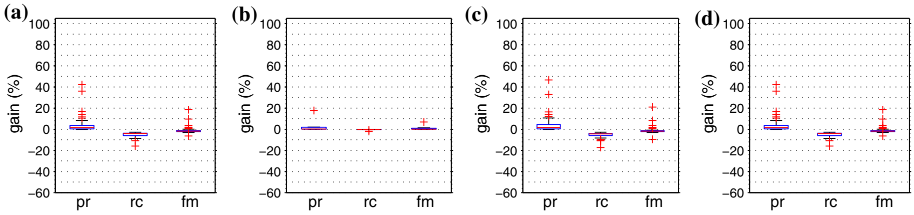

Fig. 7 Repair impact for *anatomy* track. **a** Subsumption, **b** equivalence, **c** Sub.→Eq, **d** Eq.→Sub

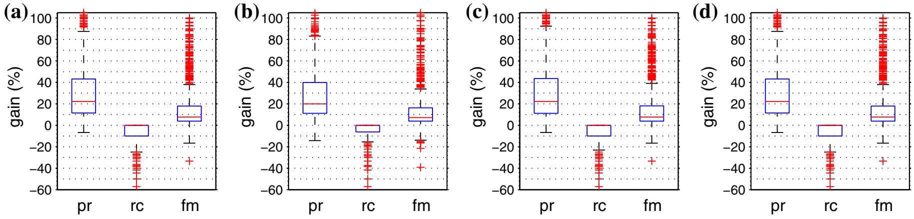

Fig. 8 Repair impact for *conference* track. **a** Subsumption, **b** equivalence, **c** Sub.→Eq, **d** Eq.→Sub

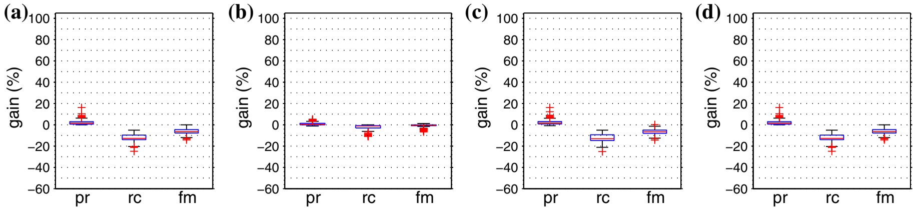

Fig. 9 Repair impact for *largebio* track. **a** Subsumption, **b** equivalence, **c** Sub.→Eq, **d** Eq.→Sub

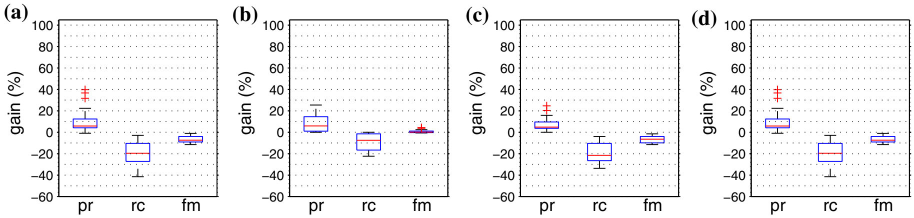

Fig. 10 Repair impact for *library* track. **a** Subsumption, **b** equivalence, **c** Sub.→Eq, **d** Eq.→Sub

allows to identify outliers (depicted as red crosses). Consider, for instance, Fig. 8a, where the box ranges approximately from 6 to 42 %, having a median value of 21% and lower (resp. upper) whisker at approximately −9 % (resp. 89 %).

Figures 7, 8, 9 and 10 confirm the general effects of a repair algorithm (i.e., an increase of precision and a decrease of recall). The results achieved by the two versions of the combined repair algorithm are totally comparable. The gain in precision results in an increase of the resulting f-measure for *conference* track and a slight decrease for *anatomy* track. However, for *largebio* and *library* tracks, the loss in f-measure is higher and lies in the range [0, 10]%.

From the results, it is evident that the effect of the combined repair algorithm is in line with that of the repair algorithm addressing exclusively subsumption violations, in terms of performance w.r.t. a reference alignment. The correlation between the repair size and its impact is again evident.

We also remind that the measures are computed against unrepaired reference alignments, while a more appropriate comparison would be against reference alignments where all the true positive conservativity violations are repaired. For instance, having the highest gain in f-measure for the *anatomy* and *conference* tracks seems to confirm our hypothesis because, in these cases, the reduced size of the input ontologies and reference alignments allows for a more effective revision, which also limits the conservativity violations (consider the reduced number of violations in Table 6a for the *conference* track) that are more frequent in the other tracks (considering of course also the difference in the size of the alignments). Finally, it is not surprising that the worst result in terms of the effect on the f-measure is obtained for the *library* track, for which the extremely high number of conservativity violations (see Table 6a) suggests the need for a more accurate revision of the reference alignment and input ontologies.

8 Related work

This section provides a general overview of the state of the art on ontology and ontology alignments debugging techniques, the related work on the *Weighted Feedback Edge Set* problem, and the notion of *assumption of disjointness*.

Conservativity principle, ontology vs alignment repair Once a violation of the conservativity principle is detected, there are different approaches to correct it by modifying the aligned ontology (i.e., targeting the input ontologies or the alignment).

The first approach is to consider as problematic all the violations and to correct them using classic ontology repair strategies on the aligned ontology, i.e., to compute a hitting set over all the justifications, as discussed at the end of Sect. 2.3. The repair computed in this way may remove elements either from the ontologies, or from the alignment, or a mixture of the two.

The second one is a family of approaches from Lambrix et al. [48,50], which targets violations that can be considered as false positives, for which the problem source is considered to stem from the incompleteness of the input ontologies. The correction strategy aims at adding to the input ontologies a minimal set of axioms, so that the input ontologies (in isolation) can entail the novel axiom (solving, in this way, the violation).

Lambrix et al. also propose a unified approach [31,49]. For each detected violation, different repair plans are generated, and they are ranked w.r.t. their informativity (i.e., considering the number of solved depending violations) and minimality (based on the number of inserted/deleted axioms). The user can classify the violations as false or true positives and can in this way influence the plan ranking. Our heuristic could be conceived as an alternative and additional plan in this multistrategy approach that supports both automatic and assisted repair.

The approaches in [31,49,50] refer to the same (simplified) context addressed here for equivalence violations, namely taxonomical projection of expressive DLs, claiming that this is one of the most used features in semantic-enabled applications.

Our proposal is orthogonal to the others discussed in this section and could be an alternative inside a multistrategy approach. However, there are settings for which the only repair target is the alignment, such as Multi-Agent Systems, where the private knowledge of each agent is encoded as a OWL ontology and cannot be modified in order to agree on a common alignment [40,41,53,60,61]. In a more general ontology matching setting, it may also be the case that one (or both) of the input ontologies should not be modified. For example, the LUCADA ontology—a medium-sized OWL ontology that describes the domain of lung cancer according to the specifications of the National Health Service (NHS) [70]—was integrated with SNOMED CT in order to facilitate interoperability with other applications within the NHS. In this use case, SNOMED CT was considered as immutable (during the matching process) since it is the reference ontology across NHS's information systems.

State-of-the-art ontology alignment repair systems, such as ALCOMO [54], AML [66], ASMOV [32], Lily [85], LogMap [33], and YAM++ [12], typically consider the input ontologies as immutable, and their repair techniques focus on the mappings.[22] Nevertheless, Pesquita et al. [62] question the automatic generation of repairs and suggest to update the ontologies when required.

Equivalence and Subsumption*violations* Ontology alignment violations are discussed in [4], where sanity checks for ontology mappings are proposed, together with general recommendations about best practices in producing ontology mappings (e.g., use of URI for identifying ontologies and matched elements). Sanity checks 6 and 7 are particularly meaningful for our analysis. Check 6 forbids the entailment of novel equivalences, among entities of the same input ontology, caused by the presence of multiple mappings sharing a common entity. This check is analysed in the context of an exclusive use of "≡" semantic relation for an alignment between an ontology of mouse anatomy and the corresponding one related to human beings (*NCI* ontology[23]). In none of the 39 detected cases, the target entities were stated equivalent in the input ontology. In a later release of such ontology, 15 entities were merged, while 18 were judged as not equivalent by domain experts (*NCI* ontology curators). This suggests that it is not possible to exclude that (new) correct equivalences may be derived using ontology matching. However, it does not seem to be the only case. Although discovering new relations between entities of the same input ontology could be very useful, their indiscriminate use,

[22] The interested reader please refer to Section 4.6.4 in [73] for an overview of these approaches.

[23] http://www.cancer.gov/cancertopics/cancerlibrary/terminologyresources.

in the absence of a manual exploratory phase, is only advisable for contexts in which correctness is not mandatory (e.g., information retrieval), while for data integration and query answering tasks, a conservative approach seems more appropriate [78]. Check 7, instead, refers to subsumption violations, for which similar considerations apply.

Another relevant approach is the one proposed by Arnold et al. [2]. Despite not being directly linked to the conservativity principle and to ontology debugging in general, the proposed algorithm could mitigate the presence of violations by refining the alignment produced by ontology matchers. The proposed method takes as input the alignment and optional background knowledge, and using a voting algorithm between different linguistic-based heuristics, it aims at refining "$\equiv$" semantic relations to "$\sqsubseteq$"/"$\sqsupseteq$" ones (and never the other way round).

(Weighted) feedback edge set problem To the best of our knowledge, no suitable heuristics have been proposed for approximating the *WFES* problem. An heuristic based on *Simulated Annealing* has been proposed by Galinier et al. [20] for solving the *FVS* and *FES* problems. Unfortunately, their approach cannot be trivially extended to *WFES*, due to their local search mechanism. On the contrary, exact and optimal approximations to the problem have been proposed [16].

Assumption of disjointness The assumption of disjointness has been originally introduced by Schlobach [67] to enhance the repair of ontologies that were underspecified in terms of disjointness axioms. In [56], a similar assumption is followed in the context of ontology mappings repair, where the authors restricted the number of disjointness axioms by using learning techniques [19,82]. These techniques, however, typically require a manually created training set. In [18], the authors present an interactive system to guide the users in the manual enrichment of the ontologies with negative constraints, including disjointness axioms.

Our proposal, as [18,19,56,82], aims at adding a small set of disjointness axioms, since adding all possible disjointness may be unfeasible for large ontologies. However, our method does not require manual intervention. Furthermore, to address the scalability problem when dealing with large ontologies and mapping sets, our method relies on the propositional projection of the input ontologies.

9 Conclusions

In this paper, we have presented a fully automatic multi-strategy method to detect and correct conservativity violations in practice. We have extended the detection and repair algorithm for subsumption violations [75] with a repair algorithm expressed by means of an *ASP* program, tailored for equivalence violations. The conducted evaluation, significantly extending that of [74], supports the practical effectiveness of our approximate methods.

Our method is sound (the violations that are detected are indeed violations if considering the full expressiveness of the input ontologies), but incomplete, since the used approximate projections of the input ontologies (i.e., Horn propositional and graph encodings) may lead to some violations being missed. In order to mitigate incompleteness, we plan to study extensions of our techniques to more expressive logical fragments, while keeping the current scalability properties. We also aim at adapting our techniques to cope with scenarios involving more than two input ontologies where new (repair) challenges arise [13].

The proposed algorithms follow a "better safe than sorry" approach, suitable for scenarios where the input ontologies are not modifiable. Nevertheless, we plan to explore alternative methods to address the conservativity violations. For example, domain experts could be involved in the assessment of the additional disjointness [18,35], and to suggest extensions to the input ontologies [31] for violations recognized as false positives.

Our techniques have been already deployed in Siemens [45] and Statoil [44] as part of BootOX [38], one of the components the "Ontology and mapping management module" provided by the Optique's platform [22]. We consider, however, that the proposed methods have also potential in scenarios others than Optique. For instance, our approach has also been successfully employed in the context of multi-agent system (MAS) [40,41] for assessing and favouring the compatibility among agents equipped with OWL 2 ontologies. The approach presented in [40,41] is based on the novel inquiry dialogue (i.e., a dialogue between two entities with the aim of solving a goal through knowledge exchange) introduced in [60,61]. The dialogue aims at reusing existing mappings related to a domain of interest, with a minimal disclosure of private knowledge.[24] In this context, on the one hand, the ontology of the other agent(s) cannot be altered, on the other hand, modifying the local ontology after any dialogue could pose some risks. For example, consider the effect of a malicious agent forging ad hoc mappings with the aim of altering the local ontology of the interlocutor. Different policies could be conceived in the dialogue. The family of approaches from Lambrix et al. [48,50] could be referred to as *optimistic*, while forbidding alterations to the local ontology as *sceptical*. Of course mixed approaches could be conceived as well. A sceptical approach, as the one followed in [40,41], could further exploit the techniques to detect violations of the consistency and conservativity principles in order to also estimate, under a game-theoretic point of view, the convenience of dealing with a particular agent, given the risk represented by these violations. In case of too many violations, the agent could simply decide to refuse to communicate and seek for another (hopefully more compatible) agent.

Another approach, described in [53], also applies ontology matching in a MAS scenario in order to allow the exchange and extension of ontology-based *action plans* among agents. In such a context, violations of the conservativity principle should be taken into account and highly critical tasks should not be performed if such violations are detected.

Acknowledgements Ernesto Jiménez-Ruiz was funded by the European Commission under FP7 Grant Agreement 318338, "Optique", and the EPSRC projects Score! ED3 and DBOnto. We also thank the invaluable help provided by Bernardo Cuenca and Ian Horrocks. We are also very grateful for the support of the Optique colleagues that facilitated our understanding of the domain, especially Dag Hovland, Evgeny Kharlamov, Dmitry Zheleznyakov, Martin Giese and Martin G. Skjæveland. Finally, we would also like to thank the anonymous reviewers of this paper.

10 Appendix

This section investigates the computational complexity of diagnosis computation ("Diagnosis Computation Complexity" in section "Appendix") and its decomposability into subproblems ("Decomposability of equivalence violations diagnosis computation" in section "Appendix").

10.1 Diagnosis computation complexity

With the aim of proving that *MAP-WFES* is *NP*-hard, Proposition 10.1 introduces a polynomial reduction from *WFES* to *MAP-WFES*, denoted as *WFES* $\preceq$ *MAP-WFES*. The intuitive idea behind the reduction is the following. Each arc (t, v, c) of the original graph is "split"

[24] Each agent is equipped with a local (private) OWL 2 ontology.

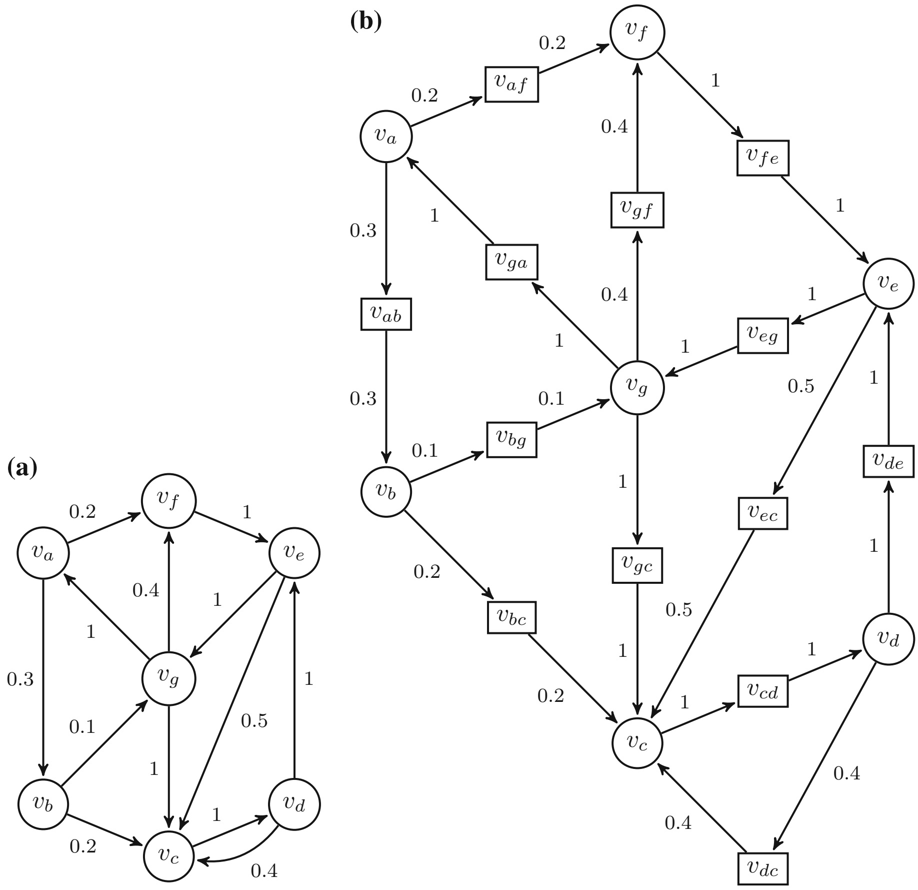

Fig. 11 Input graph G for *WFES* (left) reduced to a corresponding graph G', input for *MAP-WFES* (right). In graph G', all the *round-shaped* vertices belong to V'_1, while *square-shaped* vertices belong to V'_2. **a** Graph G (reduction input) **b** Graph G' (reduction output)

into two arcs (t, u, c) and (u, v, c), with u a fresh node. All the nodes t, v are associated with one of the input ontology, while the fresh nodes are associated with the other. In this way, all the arcs are mappings (i.e., all of them are potentially removable, exactly as the original arcs). It is easy to see that the reduction preserves the solution weight and that a 1–1 correspondence exists between cycles in the two graphs. In addition, we remark that *MAP-WFES* does not break cycles traversing only vertices of one of the input ontologies. No such cycles can exist because all the arcs are mappings, as discussed above. A reduction example is given in Example 10.1, followed by the definition of the reduction in Proposition 10.1.

Example 10.1 In Fig. 11, graphs G (left) and G' (right) are shown. *WFES* $\preceq$ *MAP-WFES* coincides with G'. The solution to G' is equal to $\Delta = \{(c, cd, 1), (b, bg, 0.1), (gf, f, 0.4), (a, af, 0.2)\}$, with a total weight of 1.7. $D = \{(b, g, 0.1), (g, f, 0.4), (a, f, 0.2), (c, d, 1)\}$ is the corresponding solution to the instance of the *WFES* problem represented by G and can be easily verified that is both minimal (having weight 1.7) and correct.

Proposition 10.1 *WFES $\preceq$ MAP-WFES. A polynomial reduction from the WFES problem to the MAP-WFES problem exists. Let $G = (V, A)$ be a digraph. The reduction consists in constructing a digraph $G' = (V', A')$ such that a subset of edges, namely $\Delta \subseteq A$, is a solution to MAP-WFES iff the corresponding set of arcs, namely D, is a solution to WFES on G. The reduction is as follows:*

1. *for each $(x, y, c) \in A$, we create a fresh vertex v_{xy}, we add it to V_2', and we create a pair of arcs (x, v_{xy}, c), (v_{xy}, y, c) that are added to A' and $\mathcal{M}$,*
2. *$V_1' = V$ and $V' = V_1' \cup V_2'$.*

A set of arcs, namely $\Delta \subseteq A'$, is a solution to the MAP-WFES problem on digraph G' iff the corresponding feedback edge set D is a solution to the WFES G, where G' is computed from G, and for each arc of the form (x, v_{xy}, c) or (v_{xy}, y, c) in Δ, we have a corresponding arc (x, y, c) in D.

Proof In order to prove the correctness of the reduction, we need to show that, if $G \preceq G'$, with G an instance of the *WFES* problem and G' an instance of the *MAP-WFES* problem, a set of arcs Δ is a (minimal) solution to G' iff the corresponding set of arcs D is a (minimal) solution to G. As discussed in [16], the proposed reduction is polynomial and it preserves graph connectivity and the weight of the solutions, by preserving a 1–1 correspondence between cycles of G and those of G', due to the 1–1 correspondency between the arcs of A and those of A'.

$\Rightarrow$: If D is a solution to *WFES* on G, Δ is a solution to *MAP-WFES* on G'. Suppose that Δ is not a solution. This requires that either at least a cycle κ' exists in digraph $(V', A' \backslash \Delta)$ or that a diagnosis Δ' exists such that $w(\Delta') < w(\Delta)$. For the first case, given the 1–1 correspondence between cycles of G and G', this implies that a corresponding cycle in G exists as well, thus contradicting that D is a solution to the instance of the *WFES* problem represented by G. For the latter case, given the 1–1 correspondence between arcs of G and G', this implies that a solution D' corresponding to Δ' exists. By the weight preservation property of the reduction, $w(D') < w(D)$ holds, contradicting that D is a (minimal) solution to the instance of the *WFES* problem represented by G.

$\Leftarrow$: If Δ is a solution to *MAP-WFES* on G', D is a solution to *WFES* on G. Suppose that D is not. This requires that either a cycle κ of G exists in digraph $(V, A \backslash D)$ or that a solution D' exists such that $w(D') < w(D)$. The first case requires the existence of a cycle κ' of G' (corresponding to κ) that is left unbroken. In turn, this either violates that Δ is a solution, or that, for some $i \in \{1, 2\}$, κ' exclusively traverses elements of V_i'. This situation is excluded by construction of G', because no arcs between vertices of the same subset V_i' of V' exist. For the latter case, this implies that a diagnosis Δ' corresponding to the (minimal) solution D' does not exist. This requires that at least an element of D' cannot belong to a diagnosis (i.e., it cannot be removed). By construction of G', we have that $\mathcal{M} = A'$. Given that only elements of $A' \backslash \mathcal{M}$ cannot belong to a diagnosis for the *MAP-WFES* problem, this results in a contradiction, thus proving the correctness of the reduction. □

From the results of Proposition 10.1, it follows that *MAP-WFES* is *NP*-hard, as detailed in Proposition 10.2.

Proposition 10.2 *MAP-WFES is NP-hard.*

Proof The proof follows from the polynomial reduction from the *WFES* problem, that is *NP*-hard [16], to *MAP-WFES*. □

10.2 Decomposability of equivalence violations diagnosis computation

Proposition 10.3 relates unsafe cycles and problematic SCCs, showing that each unsafe cycle results in a problematic SCC.

Proposition 10.3 *A SCC is problematic iff it (totally) contains at least one unsafe cycle.*

Proof $\Rightarrow$: Consider a problematic SCC S, with projections Π_1 and Π_2 on the input ontologies. From problematic SCC definition (Definition 4.6), at least one of the projections of S, say Π_1, is not a local SCC. We therefore also know that Π_1 is not a SCC; otherwise, it would also be a local SCC. Suppose that Π_1 is a subset of a SCC Π'. This implies that all the elements of Π' belongs to S as well, and therefore, Π' and Π_1 are identical, but this contradicts the assumption that Π_1 is not a SCC.

$\Leftarrow$: from the definition of cycle and SCC, each cycle κ is contained in a SCC S (i.e., $\kappa \subseteq S$). Let κ be an unsafe cycle, and let also κ_1 (resp. κ_2) be the subset of vertices of κ belonging to an input ontology $\mathcal{O}_1$ (resp. $\mathcal{O}_2$). By definition of unsafe cycle (Definition 4.3), at least one of this subsets, say κ_1, is not contained in any local SCC. But given that $\kappa \subseteq S$, $\kappa_1 \subseteq \Pi_{\mathcal{O}_1}(S)$ holds. Therefore, $\Pi_{\mathcal{O}_1}(S)$ is not contained in any local SCC either and, by Definition 4.6, S is a problematic SCC. □

Proposition 10.3 guarantees completeness for a detection technique for violations to the conservativity principle on a graph representation of an aligned ontology, based on problematic SCCs. Given that all the violations result in unsafe cycles and that they totally belong to a single problematic SCC, completeness for a repair technique breaking all the unsafe cycles follows.

Notice also that a (unsafe) cycle always belongs to one and only one (problematic) SCC (as expressed by Proposition 10.4), while a problematic SCC may contain more than one cycle. Therefore, a technique detecting problematic SCCs may be more efficient than one directly addressing unsafe cycles.

Proposition 10.4 *Safe cycle never traverse multiple SCCs of the same input ontology.*

Proof By Definition 4.3, all the vertices belonging to a projection Π of a safe cycle κ^s need to be traversed by a cycle κ' in the input ontology these vertices belong to. The claim is that cycle κ' identifies either a SCC of the aligned ontology or a subset of a SCC. Assume that vertices of Π belong to at least two SCCs S_1, S_2, that is, $\Pi \cap S_1 \neq \emptyset$ and $\Pi \cap S_2 \neq \emptyset$. Being traversed by a cycle, all the vertices of Π are mutually reachable. Then, from transitivity of reachability, it follows that all the vertices in $S_1 \cup S_2$ are mutually reachable. This contradicts the hypothesis that S_1 and S_2 are two distinct SCCs, thus proving the proposition. Such argument be can straightforwardly generalized to more than two SCCs. □

Proposition 10.5 proves the correctness of our approach and the optimality of the computed (global) diagnosis.

Proposition 10.5 *Computing a (global) diagnosis for a graph G, representing an aligned ontology, can be reduced to computing the (local) diagnoses for the problematic SCCs of G. The (minimal) global diagnosis is the union of the (minimal) local diagnoses.*

Proof From Proposition 10.3, it follows that: (i) all and only problematic SCCs contain unsafe cycles, (ii) an unsafe cycle does not traverse vertices of more than one SCC (i.e., the unsafe cycles of distinct SCCs are totally disjoint). From (i), completeness follows (it is sufficient to compute a diagnosis for each problematic SCCs to remove all the unsafe cycles in the aligned ontology). (ii) ensures the independence of SCCs, and this guarantees minimality and correctness for local diagnoses computed in isolation. Finally, it is immediate to see that the minimality property is preserved by the union of local diagnoses, and this concludes the proposition. □

Proposition 10.5 thus guarantees that the global diagnosis computed as the union of the diagnoses of the problematic SCCs is both minimal and correct (that is, it breaks all the unsafe cycles).

References

1. Agrawal R, Borgida A, Jagadish HV (1989) Efficient management of transitive relationships in large data and knowledge bases. In: ACM SIGMOD Conference on Management of Data, pp 253–262
2. Arnold P, Rahm E (2013) Semantic enrichment of ontology mappings: a linguistic-based approach. In: Advances in Databases and Information System - East European Conference (ADBIS), pp 42–55
3. Baader F, Horrocks I, Sattler U (2008) Chapter 3 description logics. In: Bruce Porter Frank van Harmelen, Vladimir Lifschitz (eds) Handbook of Knowledge Representation, vo 3 of Foundations of Artificial Intelligence, Elsevier, pp 135–179
4. Beisswanger E, Hahn U (2012) Towards valid and reusable reference alignments: ten basic quality checks for ontology alignments and their application to three different reference data sets. J Biomed Semant 3(Suppl 1):S4
5. Bodenreider O (2004) The unified medical language system (UMLS): integrating biomedical terminology. Nucleic Acids Res 32:267–270
6. Borgida A, Serafini L (2003) Distributed description logics: assimilating information from peer sources. J Data Semant 1:153–184
7. Christophides V, Plexousakis D, Scholl M, Tourtounis S (2003) On labeling schemes for the semantic web. In: International World Wide Web Conference (WWW), pp 544–555
8. Cuenca Grau B, Horrocks I, Kazakov Y, Sattler U (2008) Modular reuse of ontologies. J Artif Intell Res (JAIR) 31:273–318
9. David J, Euzenat J, Scharffe F, Trojahn C (2011) The alignment API 4.0. Semant Web J 2(1):3–10
10. Dovier A, Formisano A, Pontelli E (2007) An experimental comparison of constraint logic programming and answer set programming. AAAI 7:1622–1625
11. Dowling WF, Gallier JH (1984) Linear-time algorithms for testing the satisfiability of propositional Horn formulae. J Log Program 1(3):267–284
12. DuyHoa N, Bellahsene Z (2013) YAM++ results for OAEI 2013. In: Ontology Matching Workshop (OM), pp 211–218
13. Euzenat J (2015) Revision in networks of ontologies. Artif Intell 228:195–216
14. Euzenat J, Meilicke C, Stuckenschmidt H, Shvaiko P, Trojahn C (2011) Ontology alignment evaluation initiative: six years of experience. J Data Semant 15:158–192
15. Euzenat J, Shvaiko P (2010) Ontology matching. Springer, Heidelberg
16. Even G, Naor JS, Schieber B, Sudan M (1998) Approximating minimum feedback sets and multicuts in directed graphs. Algorithmica 20(2):151–174
17. Faria D, Pesquita C, Santos E, Palmonari M, Cruz Isabel F, Couto FM (2013) The agreementmakerlight ontology matching system. In: OTM Conferences, pp 527–541
18. Ferré S, Rudolph S (2012) Advocatus diaboli–exploratory enrichment of ontologies with negative constraints. In: International Conference on Knowledge Engineering (EKAW), pp 42–56
19. Fleischhacker D, Völker J (2011) Inductive learning of disjointness axioms. In: OTM Conferences, pp 680–697
20. Galinier P, Lemamou E, Bouzidi MW (2013) Applying local search to the feedback Vertex set problem. J Heur 1–22
21. Gallo G, Urbani G (1989) Algorithms for testing the satisfiability of propositional formulae. J Log Program 7(1):45–61
22. Giese M, Soylu A, Vega-Gorgojo G, Waaler A, Haase P, Jiménez-Ruiz E, Lanti D, Rezk M, Xiao G, Özçep Ö, Rosati R (2015) Optique: zooming in on big data. IEEE Comput 48(3):60–67
23. Glimm B, Horrocks I, Motik B, Stoilos G, Wang Z (2014) Hermit: an OWL 2 reasoner. J Autom Reason 53(3):245–269

24. Golbeck J, Fragoso G, Hartel FW, Hendler JA, Oberthaler J, Parsia B (2003) The national cancer institute's thésaurus and ontology. J Web Semant 1(1):75–80
25. Gonçalves RS, Parsia B, Sattler U (2012) Concept-based semantic difference in expressive description logics. In: International Semantic Web Conference (ISWC), Springer, pp 99–115
26. Grau BC, Horrocks I, Kazakov Y, Sattler U (2007) Just the right amount: extracting modules from ontologies. In: International Conference on World Wide Web (WWW), pp 717–726. ACM
27. Grau BC, Horrocks I, Motik B, Parsia B, Patel-Schneider PF, Sattler U (2008) OWL 2: the next step for OWL. J Web Semant 6(4):309–322
28. Horridge M (2011) Justification based explanation in ontologies. PhD thesis, University of Manchester
29. Horridge M, Parsia B, Sattler U (2008) Laconic and precise justifications in OWL. In: International Semantic Web Conference (ISWC), pp 323–338
30. Horrocks I, Kutz O, Sattler U (2006) The even more Irresistible SROIQ. In: International Conference on Principles of Knowledge Representation and Reasoning (KR), pp 57–67
31. Ivanova V, Lambrix P (2013) A unified approach for aligning taxonomies and debugging taxonomies and their alignments. In: European Semantic Web Conference (ESWC), Springer, pp 1–15
32. Jean-Mary Yves R, Patrick Shironoshita E, Kabuka Mansur R (2009) Ontology matching with semantic verification. J Web Semant 7(3):235–251
33. Jiménez-Ruiz E, Grau BC (2011) LogMap: logic-based and scalable ontology matching. In: International Semantic Web Conference (ISWC), pp 273–288,
34. Jiménez-Ruiz E, Grau BC, Horrocks I (2012) On the feasibility of using OWL 2 DL reasoners for ontology matching problems. In: OWL Reasoner Evaluation Workshop (ORE)
35. Jiménez-Ruiz E, Grau BC, Horrocks I, Berlanga R (2009) Ontology integration using mappings: towards getting the right logical consequences. In: European Semantic Web Conference (ESWC), pp 173–187
36. Jiménez-Ruiz E, Grau BC, Horrocks I, Berlanga R (2011) Logic-based assessment of the compatibility of UMLS ontology sources. J Biomed Semant 2(Suppl 1):S2
37. Jiménez-Ruiz E, Grau BC, Zhou Y, Horrocks I (2012) Large-scale interactive ontology matching: algorithms and implementation. In: European Conference on Artificial Intelligence (ECAI), pp 444–449
38. Jiménez-Ruiz E, Kharlamov E, Zheleznyakov D, Horrocks I, Pinkel C, Skjæveland MG, Thorstensen E, Mora J (2015) BootOX: practical mapping of RDBs to OWL 2. In: International Semantic Web Conference (ISWC), pp 113–132
39. Jiménez-Ruiz E, Meilicke C, Grau BC, Horrocks I (2013) Evaluating mapping repair systems with large biomedical ontologies. In: Description Logics (DL), pp 246–257
40. Jiménez-Ruiz E, Payne TR, Solimando A, Tamma V (2015) Avoiding alignment-based conservativity violations through dialogue. In: International Workshop on OWL: Experiences and Directions (OWLED)
41. Jiménez-Ruiz E, Payne TR, Solimando A, Tamma VAM (2016) Limiting logical violations in ontology alignment through negotiation. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference (KR), pp 217–226
42. Kalyanpur A, Parsia B, Horridge M, Sirin E (2007) Finding all justifications of OWL DL entailments. In: International Semantic Web Conference (ISWC), pp 267–280
43. Kazakov Y, Krötzsch M, Simancik F (2014) The incredible ELK - from polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies. J Autom Reason 53(1):1–61
44. Kharlamov E, Hovland D, Jiménez-Ruiz E, Lanti D, Lie H, Pinkel C, Rezk M, Skjæveland MG, Zheleznyakov E, Horrocks I (2015) Ontology based access to exploration data at statoil. In: International Semantic Web Conference (ISWC), pp 93–112
45. Kharlamov E, Solomakhina N, Özçep ÖL, Zheleznyakov D, Hubauer T, Lamparter S, Roshchin M, Soylu A, Watson S (2014) How semantic technologies can enhance data access at siemens energy. In: International Semantic Web Conference (ISWC), pp 601–619
46. Konev B, Walther D, Wolter F (2008) The logical difference problem for description logic terminologies. In: International Joint Conference on Automated Reasoning (IJCAR), pp 259–274
47. Kontchakov R, Wolter F, Zakharyaschev M (20084) Can you tell the difference between DL-Lite ontologies?. In: International Conference on Principles of Knowledge Representation and Reasoning (KR)
48. Lambrix P, Dragisic Z, Ivanova V (2013) Get my pizza right: repairing missing is-a relations in $\mathcal{ALC}$ ontologies. In: Semantic Technology, Springer, pp 17–32
49. Lambrix P, Liu Q (2013) Debugging the missing is-a structure within taxonomies networked by partial reference alignments. Data Knowl Eng (DKE) 86:179–205
50. Lambrix P, Wei-Kleiner F, Dragisic Z, Ivanova V (2013) Repairing missing is-a structure in ontologies is an abductive reasoning problem. In: International Workshop on Debugging Ontologies and Ontology Mappings (WoDOOM), pp 33
51. Lutz C, Walther D, Wolter F (2007) Conservative extensions in expressive description logics. Int Jt Conf Artif Intell (IJCAI) 7:453–458

52. Lutz C, Wolter F (2010) Deciding inseparability and conservative extensions in the description logic EL. J Symb Comput 45(2):194–228
53. Mascardi V, Ancona D, Barbieri M, Bordini RH, Ricci A (2014) CooL-AgentSpeak: Endowing Agentspeak-DL agents with plan exchange and ontology services. Web Intell Agent Syst 12(1):83–107
54. Meilicke C (2011) Alignments incoherency in ontology matching. PhD thesis, University of Mannheim
55. Meilicke C, Stuckenschmidt H, Tamilin A (2009) Reasoning support for mapping revision. J Log Comput 19(5):807–829
56. Meilicke C, Völker J, Stuckenschmidt H (2008) Learning disjointness for debugging mappings between lightweight ontologies. In: International Conference on Knowledge Engineering (EKAW), pp 93–108
57. Melnik S, Garcia-Molina H, Rahm E (2002) Similarity flooding: a versatile graph matching algorithm and its application to schema matching. In: IEEE International Conference on Data Engineering (ICDE), pp 117–128
58. Motik B, Grau BC, Horrocks I, Sattler U (2009) Representing ontologies using description logics, description graphs, and rules. Artif Intell J 173(14):1275–1309
59. Nebot V, Berlanga R (2009) Efficient retrieval of ontology fragments using an interval labeling scheme. Inf Sci J 179(24):4151–4173
60. Payne TR, Tamma V (2014) A Dialectical approach to selectively reusing ontological correspondences. In: Knowledge Engineering and Knowledge Management (EKAW), Springer, pp 397–412
61. Payne TR, Tamma V (2014) Negotiating over ontological correspondences with asymmetric and incomplete knowledge. In: International Conference on Autonomous Agents and Multi-Agent Systems, (AAMAS), pp 517–524
62. Pesquita C, Faria D, Santos E, Couto FM (2013) To repair or not to repair: reconciling correctness and coherence in ontology reference alignments. In: Ontology Matching Workshop (OM), pp 13–24
63. Raymond R (1987) A theory of diagnosis from first principles. Artif Intell J 32(1):57–59
64. Rodriguez-Muro M, Rezk M (2015) Efficient SPARQL-to-SQL with R2RML mappings. J Web Semant 33:141–169
65. Rosse C, Mejino JLV Jr (2003) A reference ontology for biomedical informatics: the foundational model of anatomy. J Biomed Inform 36(6):478–500
66. Santos E, Faria D, Pesquita C, Couto F (2013) Ontology alignment repair through modularization and confidence-based heuristics. arXiv:1307.5322 preprint
67. Schlobach S (2005) Debugging and semantic clarification by pinpointing. In: European Semantic Web Conference (ESWC), Springer, pp 226–240
68. Schlobach S, Cornet R (2003) Non-standard reasoning services for the debugging of description logic terminologies. In: International Joint Conference on Artificial Intelligence (IJCAI), pp 355–362
69. Schulz S, Cornet R, Spackman KA (2011) Consolidating SNOMED CT's ontological commitment. Appl Ontol 6(1):1–11
70. Sesen MB, Bañares-Alcántara R, Fox J, Kadir T, Brady JM (2012) Lung Cancer Assistant: an ontology-driven, online decision support prototype. In: International Workshop on OWL: Experiences and Directions (OWLED)
71. Shvaiko P, Euzenat J (2012) Ontology matching: state of the art and future challenges. Trans Knowl Data Eng (TKDE) 25(1):158–176
72. Skjæveland MG, Lian EH, Horrocks I (2013) Publishing the Norwegian Petroleum directorate's FactPages as semantic eeb data. In: International Semantic Web Conference (ISWC), pp 162–177
73. Solimando A (2015) Change Management in the Traditional and Semantic Web. PhD thesis, University of Genoa. https://github.com/asolimando/logmap-conservativity/raw/master/SolimandoA-thesis.pdf
74. Solimando A, Jiménez-Ruiz E, Guerrini G (2014) A multi-strategy approach for detecting and correcting conservativity principle violations in ontology alignments. In: International Workshop on OWL: Experiences and Directions (OWLED), pp 13–24
75. Solimando A, Jiménez-Ruiz E, Guerrini G (2014) Detecting and correcting conservativity principle violations in ontology-to-ontology mappings. In: International Semantic Web Conference (ISWC), pp 1–16
76. Solimando A, Jiménez-Ruiz E, Guerrini G (2015) On the feasibility of using OWL 2 reasoners in ontology alignment repair problems. In: OWL Reasoner Evaluation Workshop (ORE), pp 60–67
77. Solimando A, Jiménez-Ruiz E, Guerrini G (2016) Pushing the limits of OWL 2 reasoners in ontology alignment repair problems. Intell Artif 10:1–18
78. Solimando A, Jiménez-Ruiz E, Pinkel C (2014) Evaluating ontology alignment systems in query answering tasks. In: International Semantic Web Posters & Demonstrations Track (ISWC), pp 301–304
79. Soylu A, Giese M, Jimenez-Ruiz E, Vega-Gorgojo G, Horrocks I (2015) Experiencing OptiqueVQS: a multi-paradigm and ontology-based visual query system for end users. Univers Access Inf Soc 15(1):129–152

80. Suntisrivaraporn B, Qi G, Ji Q, Haase P (2008) A modularization-based approach to finding all justifications for OWL DL entailments. In: Asian Semantic Web Conference (ASWC), pp 1–15
81. Tarjan R (1972) Depth-first search and linear graph algorithms. SIAM J Comput 1(2):146–160
82. Völker J, Vrandecic D, Sure Y, Hotho A (2007) Learning disjointness. In: European Semantic Web Conference (ESWC), pp 175–189
83. Šváb O, Svátek V, Berka P, Rak D, Tomášek P (2005) Ontofarm: towards an experimental collection of parallel ontologies. In: International Semantic Web Conference (ISWC). Poster Session
84. W3C as Hitzler P, Krötzsch M, Parsia B, Patel-Schneider PF, Rudolph S (2009) OWL 2 Web Ontology language primer. http://www.w3.org/TR/owl2-primer/
85. Wang P, Xu B (2012) Debugging ontology mappings: a static approach. Comput Inform 27(1):21–36
86. Zhang S, Mork P, Bodenreider O (2004) Lessons learned from aligning two representations of anatomy. In: International Conference on Principles of Knowledge Representation and Reasoning (KR)
87. Zimmermann A, Euzenat J (2006) Three semantics for distributed systems and their relations with alignment composition. In: International Semantic Web Conference (ISWC). Springer, pp 16–29

Alessandro Solimando received his diplomas and PhD degree from the University of Genova (Italy). In 2015, he has joined as a postdoctoral researcher the CEDAR team, a joint laboratory between Inria Saclay and Ècole Polytechnique (France), focusing on data management and data analysis. His research currently focuses on query optimization for reformulation-based query answering over relational databases. More generally, his research interests include topics from artificial intelligence and data management, such as ontology-based data access, ontology alignment, data and schema co-evolution, and efficient query answering. Alessandro's publications include top conferences and journals such as VLDB, AAAI, KR, ISWC, and TCS.

Ernesto Jiménez-Ruiz is a postdoctoral research assistant at the Department of Computer Science, University of Oxford (UK). He obtained his PhD in Computer Science with honours from the University Jaume I (Spain), and he was awarded with the Extraordinary Doctoral Award (Engineering category). His research currently covers several areas, including biomedical information processing, ontology reuse, ontology versioning, ontology alignment, and text mining. He has published more than 80 papers in peer-reviewed journals and conferences, including 7 Quartile-1 journals, 5 Quartile-2 journals, 2 CORE-A* conference papers, and 12 CORE-A conference papers. He has also served as reviewer in a number of top-tier conferences and journals.

Giovanna Guerrini received a PhD in Computer Science from the University of Genova (Italy) where she is associate professor since 2005. Previously, she had been associate professor at the University of Pisa. Her research activity has been carried out in the database area, with specific interests in data heterogeneity, approximate retrieval, data sources integration and evolution. Previously, main research interests included object-oriented databases, XML data management, and (spatio-)temporal data handling. On these topics, she has published more than 120 papers in peer-reviewed journals and conferences. She is member of the Executive Board of the EDBT association, she has served as General Chair for EDBT/ICDT 2103, as Program Chair for ADBIS 2013, as Demo Chair for ICDE 2014, as Workshop and Tutorial Chair for ICDE 2016, and as PC member in a number of top-tier conferences.