Web Intelligence 16 (2018) 15–35
DOI 10.3233/WEB-180371
IOS Press

A graph-based approach for resolving incoherent ontology mappings

Weizhuo Li [a,b,*], Songmao Zhang [a] and Guilin Qi [c]
[a] *MADIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China*
E-mail: liweizhuo@amss.ac.cn
[b] *University of Chinese Academy of Sciences, Beijing, China*
E-mail: smzhang@math.ac.cn
[c] *School of Computer Science and Engineering, Southeast University, Nanjing, China*
E-mail: gqi@seu.edu.cn

AUTHOR COPY

Abstract. Ontology mappings are regarded as the semantic bridges that link entities from different yet overlapping ontologies in order to support knowledge sharing and reuse on the Semantic Web. However, mappings can be wrong and result in logical conflicts among ontologies. Such kind of mappings are called incoherent mappings. As an important part of ontology matching, mapping validation aims at detecting the conflicts and restoring the coherence of mappings. In this paper, we propose a graph-based approach which is complete for detecting incoherent mappings among DL-Lite ontologies. The lightweight DL-Lite family of description logics stand out for tractable reasoning and efficient query answering capabilities. Our approach consists of a set of graph construction rules, a graph-based incoherence detection algorithm, and a graph-based incoherence repair algorithm. We propose and formalize three repair principles in an attempt to measure the wrong mappings, where the notion of common closures w.r.t. a mapping arc in the constructed graph is introduced. These principles feature a global removal strategy that is independent of individual ontology matchers. In order to relieve the loss of information among ontologies in the repair process, we further define a mapping revision operator so that common closures related to the removed mappings can be preserved in the graph. We implement the graph-based algorithms and evaluate their performance in a comparison with state-of-the-art systems on real-world ontologies. Experimental results show that our approach can remove more wrong mappings and achieve better repairing results in most of the cases.

Keywords: Semantic Web, ontology alignment, mapping validation, alignment repairing, incoherence

1. Introduction

In order to share and reuse knowledge on the Semantic Web, more and more domain knowledge is encoded in the form of ontology [1]. Mappings are regarded as the semantic bridges that link entities from different yet overlapping ontologies so as to support the communication of applications built upon these ontologies. Many promising matchers have been developed to generate mappings across ontologies in (semi-)automatic ways [2]. One of the challenges in managing such interlinked dynamic knowledge is to handle the potential incoherences caused by introducing mappings [3]. Incoherent mappings can derive unsatisfiable concepts and properties and have negative impacts on applications on the Semantic Web including terminological reasoning, data transformation, P2P collaboration and query answering [4,5]. Mapping validation or alignment repairing, which plays an important role in ontology matching, aims at detecting the conflicts and restoring the coherence of mappings [6].

Up to now, the mainstream methods of mapping validation interpret mappings as sets of axioms in description logics (DL) or restricted logic programs, and use logical reasoners to detect the incoherence. For in-

*Corresponding author. E-mail: liweizhuo@amss.ac.cn.

2405-6456/18/$35.00 © 2018 – IOS Press and the authors. All rights reserved

stance, such technique has been used in Alcomo [5], LogMap [7], etc. In the repair stage, they adopt some local or global strategies to remove mappings so as to regain the coherence. Alternatively, there exist some works that consider the uncertainty of mappings and model mapping validation as an optimization problem and employ probabilistic reasoning techniques to solve the conflicts such as ContraBovemRufum [8] and ELog [9].

Although can regain the coherence, the state-of-the-art methods often suffer from two limitations. First, most methods rely on the weights of mappings as a measure of correctness of the mappings in conflict sets. Such weights are assigned or computed by individual ontology matchers, thus they can be unreliable and have negative impacts on the quality of mapping repairing. Moreover, when mappings are removed to regain coherence, the knowledge implied in these mappings are discarded at the same time, which could be potentially exploited for indicating entailments across ontologies [10]. In the domain of ontology revision, there are works that employ revision operators to preserve the knowledge implied by the deleted axioms as long as it is consistent with the repaired ontology [3,11–14]. Inspired by these works, we attempt to explore a finer-grained approach to resolve incoherent mappings by applying revision operators.

More specifically, we focus on repairing the incoherent mappings among DL-Lite ontologies. The family of DL-Lite ontologies stands out for their tractable reasoning and efficient query answering capabilities [15], where the problem of TBox classification can be reduced to computing the transitive closure of a graph [16]. Accordingly, we propose a graph-based approach for mapping validation. First, ontologies and mappings are encoded into a directed graph by the construction rules we designed, where subsumption relations between concepts or roles can be derived from reachability of nodes. To detect incoherent mappings, we define minimal incoherent path pairs (MIPPs) in the constructed graph, which correspond to the minimal incoherence preserving subsets (MIPSs) [5] in the state-of-the-art mapping validation methods, and MIPPs can be computed by backtracking the pairs of disjoint nodes in the graph. In the repair stage, in order to remove the mappings that are actually wrong, we introduce the notion of common closures w.r.t. a mapping arc in the constructed graph, as a measure of how many the mapping has in common with other mappings on supporting the same linkages across ontologies. We propose three elaborated repair principles, and define functions that fulfill these principles. To relieve the loss of information, a mapping revision operator is designed based on these functions, so that the implied knowledge of the removed mappings can be restored when no incoherence is caused.

The contribution of this paper is summarized as follows.

1. We develop a graph-based approach for mapping validation, which is complete for detecting the incoherence of mappings among DL-Lite ontologies. The approach consists of a set of graph construction rules, a graph-based incoherence detection algorithm, and a graph-based incoherence repair algorithm.
2. We propose and formalize three repair principles in an attempt to measure the wrong mappings, where the notion of common closures w.r.t. a mapping arc in the constructed graph is introduced. These principles feature a global removal strategy that is independent of individual ontology matchers. In order to relieve the loss of information among ontologies in the repair process, we further define a mapping revision operator so that common closures related to the removed mappings can be preserved in the graph.
3. We implement the graph-based algorithms and evaluate their performance in a comparison with state-of-the-art systems on real-world ontologies. Experimental results show that our approach can remove more wrong mappings and achieve better repairing results in most of the cases.

The rest of this paper is organized as follows. Related works are introduced in Section 2. In Section 3, we introduce the theoretical basis of DL-Lite and ontology mapping. Section 4 presents the graph construction method. The detection algorithm and repair algorithm based on the constructed graph are proposed in Section 5 and Section 6, respectively. The comprehensive evaluation of our approach is presented in Section 7, followed by a conclusion in Section 8.

2. Related work

In this section, we review the research efforts on mapping validation in three aspects as follows.

2.1. Repair methods based on logical entailments

A comprehensive framework for alignment repair based on diagnosis theory was developed in [5,17–

AUTHOR COPY

19]. Minimal incoherence preserving subalignments (MIPS) and minimal unsatisfiability preserving subalignments (MUPS) were defined as minimal sets of mappings generating inconsistency or incoherence. ALCOMO (Applying Logical Constraints On Matching Ontologies) is the direct result of these works [5]. Rather than an ontology matcher, it provides a library of tools for computing MIPS, MUPS, and diagnoses in either a complete or constraint-based way. It thus allows for implementing measures independent from individual matchers [6].

Qi et al. defined a relevance-based selection function to describe the relevance among axioms within ontology, and provided the corresponding iterative algorithm for mapping validation [3]. Three concrete ontology revision operators were designed to instantiate the iterative algorithm, resulting in three different mapping revision algorithms. Moreover, they showed that the algorithm given in [17] can be encoded as a special iterative algorithm.

The ASMOV system detects inconsistency through anti-patterns and corrects the alignment before its final delivery [20]. Instead of using a complete solver, the system recognizes sets of correspondences that are proved to lead to an inconsistency. The semantic verification process examines five types of patterns e.g., disjoint-subsumption contradiction, or subsumption incompleteness [6].

The LogMap system uses a logical reasoner in order to pinpoint inconsistencies and incoherent classes [7]. In order to scale to large ontologies, it uses a Horn propositional logic representation of the extended hierarchy of each ontology together with all existing mappings and employs an Dowling–Gallier algorithm [21] to model propositional Horn satisfiability. In order to keep mappings coherent, it removes the mappings with the lowest confidence among the smallest sets of mappings that cause the inconsistency.

Santos et al. proposed a repair algorithm tailored for large biomedical ontologies, called AMLR [22]. They introduced a modularization based technique to extract the core fragments of the ontologies that contained solely the necessary classes and relations caused by disjoint restrictions, and utilized confidence-based heuristics to determine near-optimal solutions for incoherent alignment.

The main concern of ALCOMO and Qi et al.' works is the scalability. Although independent of specific logical systems, they sacrifice efficiency when employ DL reasoners to detect and resolve the incoherence, thus may not be suitable for large-scale repair tasks. The inconsistency patterns used by ASMOV are semantically correct, but not complete, so that the mappings after repair may still be incoherent. Moreover, it may reject mappings involved in their patterns based on the confidence when iterating the matching process [6]. LogMap is efficient and scalable, but its coarse-grained strategy and approximated transformation cannot guarantee the complete results. AMLR is one of the best repair systems tailored for large biomedical ontologies. Nevertheless, It was an incomplete method as only unsatisfied classes due to disjointness conflicts were considered.

2.2. Repair methods based on probabilistic reasoning

Another type of repair methods incorporates the confidence or similarity of mappings and employs probabilistic reasoning techniques, of which ContraBovemRufum [23] and ELog [9] are two representatives.

Due to the fact that standard satisfiability checking regarding the underling DL can be reduced to probabilistic reasoning problems [24], ContraBovemRufum extended P-$\mathcal{SHIQ}(\mathcal{D})$ based on $\mathcal{SHIQ}(\mathcal{D})$ and transformed mappings into conditional constraints by translation rules. Although the repair method of ContraBovemRufum is based on a greedy strategy, it is only applicable to small-scaled repair tasks because of the increase in complexity caused by probabilistic reasoning techniques.

ELog is a reasoner for log-linear description logics and offers complete reasoning capabilities for $\mathcal{EL}^{++}$ [9]. The debugging problem is modeled to find the maximum a-posteriori (MAP) state of a Markov Logic Network, which corresponds to an optimal solution. ELog is heavily dependent on weights of mappings, so it may not be competent for qualitative matchers such as StringEquiv and FCA-Map.

The probabilistic reasoning based methods normally do not have a good interpretability [25], thus difficult for humans to comprehend the reason of incoherence and the rational of the repair.

2.3. Uncertainty languages for mapping validation

The probabilistic extensions of various Web languages for representing ontology mappings [26,27] can be conceivably applied to mapping validation, for instance, Probabilistic Description Logic Programs [28] and Possibilistic Logics [29].

AUTHOR COPY

In order to deal with the uncertainty and inconsistency in mappings, Lukasiewicz et al. proposed a tightly integrated approach to probabilistic disjunctive description logic programs [30], which tightly combined normal logic programs under the well-founded answers with both tractable ontology languages and Bayesian probabilities. They showed that this language could be used to resolve inconsistencies and merge mappings from different matchers based on the level of confidence assigned by various rules.

Qi et al. proposed the Possibilistic Logics as a possibilistic extension of DLs [29], in which inference services could be reduced to the task of computing the inconsistency degree of a knowledge base in possibilistic description logics. For the drowning problem in possibilistic inference, authors adapted the linear order inference to deal with the problem in possibilistic logics. They also showed the application of the logics in ontology merging with incoherent mappings.

All these works are theoretical frameworks exploring novel repair strategies for solving incoherent mappings. No algorithms or implemented systems are provided yet.

3. Preliminaries

In this section, first we briefly recall the DL-Lite family, and then introduce ontology mappings and their incoherence.

3.1. The DL-Lite family

We start with the introduction of DL-Lite$_{\text{core}}$, which is the core language for the DL-Lite family [31]. DL-Lite$_{\text{core}}$ has pairwise disjoint sets of concepts, roles and individuals, where concepts and roles are further partitioned into atomic, basic, and general sets, respectively. In our convention, A denotes an atomic concept, P an atomic role, B a basic concept, Q a basic role, C a general concept, and R a general role. Moreover, $\bot$ denotes the bottom concept. In DL-Lite$_{\text{core}}$, a basic role is either an atomic role or the inverse of an atomic role, and a basic concept can be an atomic concept or a concept of the form $\exists Q$. Moreover, a general concept is either a basic concept or the negation of a basic concept, whereas a general role is a basic role or the complement of a basic role. Syntactically, the DL-Lite$_{\text{core}}$ concepts and roles are defined as follows: (1) $C := B|\neg B$, (2) $B := A|\exists Q$, (3) $Q := P|P^-$, and (4) $R := Q|\neg Q$.

An axiom is an expression taking one of the following forms: (1) the concept inclusion axiom $B \sqsubseteq C$, (2) the role inclusion axiom $Q \sqsubseteq R$, (3) the functionality axiom funct(Q), (4) the concept membership axiom $A(a)$, where a is an individual, or (5) the role membership axiom $P(a, b)$, where a and b are individuals. In DL-Lite$_{\text{core}}$, an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$, where $\mathcal{T}$ is a finite set of concept inclusion axioms and $\mathcal{A}$ a finite set of membership axioms of concepts and roles. Further, DL-Lite$_{\mathcal{R}}$ extends DL-Lite$_{\text{core}}$ with role inclusion axioms and DL-Lite$_{\mathcal{F}}$ allows for functionality axioms based on DL-Lite$_{\text{core}}$.

The semantics of the DL-Lite logics is based on the general first-order interpretation [32]. An interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ consists of a non-empty set $\Delta^{\mathcal{I}}$, called the domain, and an interpretation function $\cdot^{\mathcal{I}}$ that associates individuals, concepts and roles to the elements of the domain, subsets of the domain, and binary relations on the domain, respectively. The interpretation function can be extended to arbitrary concept and role descriptions and axioms in a standard way [31].

The satisfaction of an axiom F in an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, denoted as $\mathcal{I} \models F$, is defined as follows: (1) $\mathcal{I} \models B \sqsubseteq C$ iff $B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$; (2) $\mathcal{I} \models Q \sqsubseteq R$ iff $Q^{\mathcal{I}} \subseteq R^{\mathcal{I}}$; (3) $\mathcal{I} \models \text{Funct}(Q)$ iff $Q^{\mathcal{I}}$ is functional; (4) $\mathcal{I} \models A(a)$ iff $a^{\mathcal{I}} \in A^{\mathcal{I}}$; (5) $\mathcal{I} \models P(a, b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$. An ontology $\mathcal{O}$ is consistent iff there exists an interpretation $\mathcal{I}$ such that for every axiom $F \in \mathcal{O}$, $\mathcal{I} \models F$ holds; and such an interpretation is called a model of $\mathcal{O}$. If axiom F is satisfied by all the models of ontology $\mathcal{O}$, we can also write $\mathcal{O} \models F$. Satisfiability of concepts and roles are specified in the following definition, which leads to the coherence of ontology.

Definition 1 ([5] Unsatisfiability and Incoherence)**.** Let $\mathcal{O}$ be a DL-Lite ontology. A concept C (role R) in $\mathcal{O}$ is unsatisfiable iff $\mathcal{O} \models C \sqsubseteq \bot$ ($\mathcal{O} \models R \sqsubseteq \bot$) holds. Ontology $\mathcal{O}$ is incoherent iff there exists at least one unsatisfiable concept or role in $\mathcal{O}$.

Note that the consistency and coherence of an ontology do not coincide. An ontology can be consistent, i.e., has a model, and incoherent at the same time, i.e., has unsatisfiable concepts or roles; and all concepts and roles being satisfiable does not imply the consistency of the ontology.

The limited expressivity of DL-Lite enables powerful computational performance. As Lembo et al. showed in [16], the problem of TBox classification in

AUTHOR COPY

a DL-Lite ontology can be solved by computing the transitive closure of a directed graph. Given a DL-Lite ontology $\mathcal{O}$ with TBox $\mathcal{T}$ over a signature $\Sigma_\mathcal{T}$ containing symbols for atomic concepts and atomic roles, the way of constructing directed graph $\mathcal{G}_\mathcal{T} = (N, E)$ from $\mathcal{T}$ over $\Sigma_\mathcal{T}$ is described as follows:

- R1: For each atomic concept A in $\Sigma_\mathcal{T}$, N contains the node A.
- R2: For each atomic role P in $\Sigma_\mathcal{T}$, N contains the nodes P, P^-, $\exists P$, and $\exists P^-$.
- R3: For each concept inclusion axiom $B_1 \sqsubseteq B_2 \in \mathcal{T}$, E contains the arc $\langle B_1, B_2\rangle$.
- R4: For each concept inclusion axiom $B_1 \sqsubseteq \neg B_2 \in \mathcal{T}$, E contains the arcs $\langle B_1, \neg B_2\rangle$ and N contains the node $\neg B_i$.
- R5: For each role inclusion axiom $P_1 \sqsubseteq P_2 \in \mathcal{T}$, E contains the arcs $\langle P_1, P_2\rangle$, $\langle P_1^-, P_2^-\rangle$, $\langle \exists P_1, \exists P_2\rangle$, and $\langle \exists P_1^-, \exists P_2^-\rangle$.
- R6: For each role inclusion $P_1 \sqsubseteq \neg P_2 \in \mathcal{T}$, E contains the arcs $\langle P_1, \neg P_2\rangle$, $\langle P_1^-, \neg P_2^-\rangle$, $\langle \exists P_1, \exists \neg P_2\rangle$, and $\langle \exists P_1^-, \neg \exists P_2^-\rangle$, and N contains the nodes $\neg P_2$, $\neg P_2^-$, $\exists \neg P_2$, and $\exists \neg P_2^-$.

Of note, extra nodes (e.g., P^-, $\exists P$, $\exists P^-$) and arcs (e.g., $\langle P_1^-, P_2^-\rangle$, $\langle \exists P_1, \exists P_2\rangle$, $\langle \exists P_1^-, \exists P_2^-\rangle$) are added in order to ensure that all the information of TBox can be preserved in the graph.

Definition 2 ([16] Directed Path)**.** In a directed graph, a directed path is a sequence of arcs where all the arcs are directed in the same direction, denoted as $S_1 \rightarrow S_2 \rightarrow \cdots \rightarrow S_t$ where $S_1, \ldots, S_t$ are nodes.

Definition 3 ([16] Transitive Closure)**.** The transitive closure of a graph $\mathcal{G} = (N, E)$ is a graph (N, E^*) such that there is an arc $\langle s, t\rangle$ in E^* iff there is a path from the node s to the node t in $\mathcal{G}$.

For a DL-Lite ontology, the arcs of the transitive closure of the graph constructed by rules R1–R6 actually represent the complete inclusion axioms, asserted and implied in its TBox. Computing such a transitive closure can be done in polynomial time [33], demonstrating the reasoning efficiency of DL-Lite ontologies.

3.2. Ontology mappings and their incoherence

We define ontology mappings in the same way as the ontology matching community do [6]. And we always assume that the ontologies to be matched are consistent and coherent. A set of mappings, often identified by one matching system or algorithm, is called an alignment.

Definition 4 ([6] Ontology Mapping)**.** Let O_i and O_j be two DL-Lite ontologies. A mapping is a 4-tuple (e_i, e_j, r, n), where e_i and e_j are two elements, i.e., concepts, roles, or individuals, from O_i and O_j, respectively, $r \in \{\sqsubseteq, \sqsupseteq, \equiv\}$ is a relation holding between e_i and e_j, and n is a weight in range [0, 1].

When the weights are ignored, the mappings together with their source ontologies can be seen as a DL-Lite ontology. Such an ontology can be incoherent caused by the introduction of the mappings across source ontologies.

Definition 5 ([5] Mapping Incoherence)**.** Given two DL-Lite ontologies O_i and O_j and the set of their mappings $\mathcal{M}$, $\mathcal{M}$ is incoherent with regard to O_i and O_j if there exists at least one concept C_k or role R_k, $k \in \{i, j\}$, such that it is satisfiable in O_k but unsatisfiable in $O_i \cup O_j \cup \mathcal{M}$ where the weights of mappings are ignored; otherwise $\mathcal{M}$ is coherent.

In the examples in this paper, subscripts are used to distinguish the entities from different ontologies.

Example 1. Ontologies O_1 and O_2 describe the domain of conference management systems, whose axioms are listed as follows:

$\text{Reviewer}_1 \sqsubseteq \text{ConferenceMember}_1$

$\text{ProgramCommitee}_1 \sqsubseteq \text{ConferenceMember}_1$

$\text{Conference}_1 \sqsubseteq \neg\text{ConferenceMember}_1$

$\text{Review}_2 \sqsubseteq \text{Document}_2$

$\text{Reviewer}_2 \sqsubseteq \text{Person}_2$

$\text{Chairman}_2 \sqsubseteq \text{Person}_2$

$\text{PC-Chair}_2 \sqsubseteq \text{Chairman}_2$

$\text{ConferenceChair}_2 \sqsubseteq \text{Chairman}_2$

$\text{Document}_2 \sqsubseteq \neg\text{Person}_2$

Their alignment $\mathcal{M}$ consists of the following mappings:

$(\text{Reviewer}_1, \text{Review}_2, \equiv, 0.8)$

$(\text{Reviewer}_1, \text{Reviewer}_2, \equiv, 0.9)$

$(\text{Chair}_1, \text{PC-Chair}_2, \equiv, 0.5)$

$(\text{Chair}_1, \text{ConferenceChair}_2, \equiv, 0.7)$

AUTHOR COPY

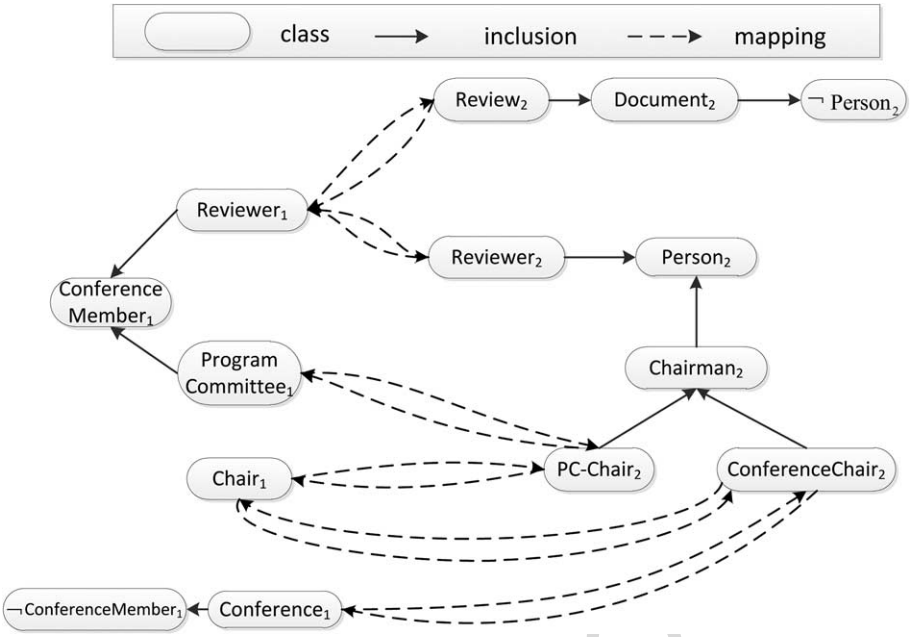

Fig. 1. The integrated graph constructed by MR1–MR6 for Example 1.

$(\text{Conference}_1, \text{ConferenceChair}_2, \equiv, 0.6)$

$(\text{ProgramCommitee}_1, \text{PC-Chair}_2, \equiv, 0.8)$

According to Definition 5, concepts $Reviewer_1$ and $Chair_1$ are unsatisfiable in $O_1 \cup O_2 \cup \mathcal{M}$ because they are subsumed by the disjoint concepts. For example, both $O_1 \cup O_2 \cup \mathcal{M} \models Reviewer_1 \sqsubseteq \neg Person_2$ and $O_1 \cup O_2 \cup \mathcal{M} \models Reviewer_1 \sqsubseteq Person_2$ hold, so $O_1 \cup O_2 \cup \mathcal{M} \models Reviewer_1 \sqsubseteq \bot$, meaning that $Reviewer_1$ is unsatisfiable, thus $\mathcal{M}$ is incoherent.

In DL-Lite, the concept and role membership axioms asserted in ABox can solely take the form of $A(a)$ and $P(a, b)$ where A is an atomic concept and P an atomic role. This can not cause any unsatisfiability, thus in the remainder of the paper, we only consider the TBox of DL-Lite ontology.

4. Constructing graphs to represent ontology mappings

Inspired by the works in [13,14], we extend graph construction rules R1–R6 and encode ontology mappings into a graph. Concretely, suppose we have graphs $\mathcal{G}_{\mathcal{T}_i} = (N_i, E_i)$ and $\mathcal{G}_{\mathcal{T}_j} = (N_j, E_j)$ from DL-Lite ontologies O_i and O_j, respectively. The mapping construction rules MR1–MR6 tailored for mappings between O_i and O_j are presented as follows where the weights of mappings are ignored:

- MR1: For each concept mapping $(C_i, C_j, \sqsubseteq)$, add an arc $\langle C_i, C_j \rangle$.
- MR2: For each concept mapping $(C_j, C_i, \sqsupseteq)$, add an arc $\langle C_j, C_i \rangle$.
- MR3: For each concept mapping $(C_i, C_j, \equiv)$, add two arcs $\langle C_i, C_j \rangle$ and $\langle C_j, C_i \rangle$.
- MR4: For each property mapping $(R_i, R_j, \sqsubseteq)$, add four arcs $\langle R_i, R_j \rangle$, $\langle R_i^-, R_j^- \rangle$, $\langle \exists R_i, \exists R_j \rangle$ and $\langle \exists R_i^-, \exists R_j^- \rangle$.
- MR5: For each property mapping $(R_j, R_i, \sqsupseteq)$, add four arcs $\langle R_j, R_i \rangle$, $\langle R_j^-, R_i^- \rangle$, $\langle \exists R_j, \exists R_i \rangle$ and $\langle \exists R_j^-, \exists R_i^- \rangle$.
- MR6: For each property mapping $(R_i, R_j, \equiv)$, add eight arcs $\langle R_i, R_j \rangle$, $\langle R_j, R_i \rangle$, $\langle R_i^-, R_j^- \rangle$, $\langle R_j^-, R_i^- \rangle$, $\langle \exists R_i, \exists R_j \rangle$, $\langle \exists R_j, \exists R_i \rangle$, $\langle \exists R_i^-, \exists R_j^- \rangle$ and $\langle \exists R_j^-, \exists R_i^- \rangle$.

The constructed graph, called the integrated graph, is denoted as $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$, where $N = N_i \cup N_j$ and $E = E_i \cup E_j$ are the union of nodes and arcs in subgraphs $\mathcal{G}_{\mathcal{T}_i}$ and $\mathcal{G}_{\mathcal{T}_j}$, respectively, and $E_{\mathcal{M}}$ represents a set of mapping arcs which act as bridges linking nodes across $\mathcal{G}_{\mathcal{T}_i}$ and $\mathcal{G}_{\mathcal{T}_j}$.

Applying MR1–MR6, the integrated graph constructed for Example 1 is shown in Fig. 1. It is a directed graph, where solid arrows represent the inclu-

AUTHOR COPY

sion axioms within the source DL-Lite ontologies and dashed arrows represent the mappings between ontologies.

Given ontologies O_i and O_j and their mappings $\mathcal{M}$, the integrated graph becomes a graph for ontology $O_i \cup O_j \cup \mathcal{M}$ constructed by R1–R6 and MR1–MR6. We can thus similarly prove that computing the transitive closure of the integrated graph can obtain the complete inclusion axioms in $O_i \cup O_j \cup \mathcal{M}$, as shown in the following theorem.

In DL-Lite, the assertions in the form $B_1 \sqsubseteq B_2$ or $R_1 \sqsubseteq R_2$ are called positive inclusions (PIs) and those like $B_1 \sqsubseteq \neg B_2$ or $R_1 \sqsubseteq \neg R_2$ are negative inclusions (NIs).

Theorem 1. *Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from DL-Lite ontologies O_i, O_j and their mappings $\mathcal{M}$, and $\mathcal{G}^* = (N, (E \cup E_{\mathcal{M}})^*)$ be the transitive closure of $\mathcal{G}$. Let S_i be a basic concept (or role) in O_i and S_j a general concept (or role) in O_j. $O_i \cup O_j \cup \mathcal{M} \models S_i \sqsubseteq S_j$ iff arc $\langle S_i, S_j \rangle \in (E \cup E_{\mathcal{M}})^*$.*

Proof. (Sketch) We follow the proof in [14] (Theorem 5).

($\Rightarrow$) If arc $\langle S_i, S_j \rangle \in (E \cup E_{\mathcal{M}})^*$, according to the definition of transitive closure of graph, there exists at least one path starting from node S_i to node S_j in $\mathcal{G}$, which corresponds to a set of inclusions and mappings in $O_i \cup O_j \cup \mathcal{M}$ according to construction rules R1–R6 and MR1–MR6. Therefore, $O_i \cup O_j \cup \mathcal{M} \models S_i \sqsubseteq S_j$ holds.

($\Leftarrow$) If $O_i \cup O_j \cup \mathcal{M} \models S_i \sqsubseteq S_j$, we consider the following two cases.

Case 1 If the negation constructor '$\neg$' does not exist in S_j, we can easily infer arc $\langle S_i, S_j \rangle \in (E \cup E_{\mathcal{M}})^*$ according to [16] (Theorem 1)

Case 2 If the negation constructor '$\neg$' exists in S_j, according to [14] (Theorem 4), there exists a minimal set including some PIs, one NI and some mappings, denoted by K, such that $K \models S_i \sqsubseteq S_j$ and $K \subseteq O_i \cup O_j \cup \mathcal{M}$. Without loss of generality, we assume that the one NI is $S' \sqsubseteq S_j$. According to Case 1, we can infer arc $\langle S_i, S' \rangle \in (E \cup E_{\mathcal{M}})^*$. Therefore, arc $\langle S_i, S_j \rangle \in (E \cup E_{\mathcal{M}})^*$ holds.

□

Theorem 1 indicates that the entailments between concepts (roles) can be reduced to the graph reachability problem. Based on this, we define notions for detecting unsatisfiable concepts and roles in the integrated graph.

Definition 6 (Path-Unsatisfiability)**.** Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i and O_j and their mappings $\mathcal{M}$. A node $S \in N$ is path-unsatisfiable if there exist two paths in $\mathcal{G}$ starting from S to node S' and to node $\neg S'$, respectively. S is called aligned path-unsatisfiable if the two paths contain at least one mapping arc. Graph $\mathcal{G}$ is incoherent iff there exists at least one path-unsatisfiable node in $\mathcal{G}$.

One can easily see that path-unsatisfiable nodes correspond to unsatisfiable concepts and roles in $O_i \cup O_j \cup \mathcal{M}$, and aligned path-unsatisfiable nodes reveal the unsatisfiability caused by mappings. The following lemma holds obviously.

Lemma 1. *If node S is aligned path-unsatisfiable with regard to an integrated graph $\mathcal{G}$, then for any arc $\langle S', S \rangle$ in the transitive closure of $\mathcal{G}$, node S' is also aligned path-unsatisfiable.*

As stated previously, we always assume that the source ontologies to be mapped are coherent and consistent, so the incoherence can only be derived from the mappings. This means that we solely need to focus on the nodes that are aligned path-unsatisfiable in the integrated graph.

Theorem 2. *Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i and O_j and their mappings $\mathcal{M}$. When both O_i and O_j are coherent and consistent, $\mathcal{M}$ is incoherent iff there exists at least one aligned path-unsatisfiable node in $\mathcal{G}$.*

Proof. ($\Rightarrow$) If there exists an aligned path-unsatisfiable node S in $\mathcal{G}$, according to Definition 6, we assume there exist two paths, one is $S \rightarrow S'$ and the other one is $S \rightarrow \neg S'$. By Theorem 1, we can infer both $S \sqsubseteq S'$ and $S \sqsubseteq \neg S'$. Therefore, $O_i \cup O_j \cup \mathcal{M}$ contains at least one unsatisfiable concept or role, and $\mathcal{M}$ is incoherent.

($\Leftarrow$) If $\mathcal{M}$ is incoherent, then there will be at least one unsatisfiable concept or role in $O_i \cup O_j \cup \mathcal{M}$. Without loss of generality, suppose C is an unsatisfiable concept and $O_i \cup O_j \cup \mathcal{M} \models C \sqsubseteq D$, $O_i \cup O_j \cup \mathcal{M} \models C \sqsubseteq \neg D$, According to Theorem 1, we have $\langle C, D \rangle \in (E \cup E_{\mathcal{M}})^*$ and $\langle C, \neg D \rangle \in (E \cup E_{\mathcal{M}})^*$. It is easy to see that exist two paths starting from C to node D and to node $\neg D$, respectively. As stated previously assumption that the source ontologies to be mapped are coherent and consistent. Therefore, this node is aligned path-unsatisfiable. □

AUTHOR COPY

5. Detecting incoherent mappings

In this section, we present a graph-based algorithm for detecting incoherent mappings. Before that, we first introduce the notion of minimal incoherent path pair (MIPP) which is the core for incoherence detection in the integrated graph.

5.1. Minimal incoherent path pairs

To resolve the incoherent alignment, normally the so-called Minimal Incoherence Preserving Subsets (MIPSs) [5] are identified with regard to ontologies and mappings. We give the definition of MIPS as follows, where the equivalent mapping between e_i and e_j cross ontologies is treated as two inclusion axioms $(e_i, e_j, \sqsubseteq)$ and $(e_j, e_i, \sqsubseteq)$ (also called mapping axioms), slightly different from the MIPS defined in [5] which represents the equivalent mapping as an equivalent axiom.

Definition 7 (Minimal Incoherence Preserving Subset)**.** Let O_i and O_j be two DL-Lite ontologies and $\mathcal{M}$ a set of their mappings. A subset $\mathcal{S} \subseteq O_i \cup O_j \cup \mathcal{M}$ is a incoherence preserving subset of $O_i \cup O_j \cup \mathcal{M}$ if $\mathcal{S}$ is incoherent. $\mathcal{S}$ is minimal if every subset $\mathcal{S}' \subset \mathcal{S}$ is coherent.

Since the incoherence is always caused by the mappings, in every MIPS, removing one mapping axiom can solve the incoherence, and solving all MIPSs can regain the satisfiability of all concepts and roles.

Example 2 (Example 1 continued)**.** There exist eight MIPSs for $O_1 \cup O_2 \cup \mathcal{M}$ in Example 1, two of which are listed as follows, which respectively cause concepts Reviewer_1 and Chair_1 to become unsatisfiable. $\{(\text{Reviewer}_1, \text{Reviewer}_2, \sqsubseteq), (\text{Reviewer}_1, \text{Review}_2, \sqsubseteq), (\text{Reviewer}_2 \sqsubseteq \text{Person}_2), (\text{Review}_2 \sqsubseteq \text{Document}_2), (\text{Document}_2 \sqsubseteq \neg\text{Person}_2)\}$, $\{(\text{Chair}_1, \text{ConferenceChair}_2, \sqsubseteq), (\text{ConferenceChair}_2, \text{Conference}_1, \sqsubseteq), (\text{Chair}_1, \text{PC-Chair}_2, \sqsubseteq), (\text{PC-Chair}_2, \text{ProgramCommitee}_1, \sqsubseteq), (\text{ProgramCommitee}_1 \sqsubseteq \text{ConferenceMember}_1), (\text{Conference}_1 \sqsubseteq \neg\text{ConferenceMember}_1)\}$.

Parallel in the integrated graph, there exist sets of paths corresponding to MIPSs that give rise to the incoherence of mappings, and we encode them as minimal incoherent path pairs, defined as follows.

Definition 8 (Minimal Incoherent Path Pair)**.** In the integrated graph constructed from two DL-Lite ontologies and their mappings, the pair of paths of an aligned path-unsatisfiable node that reach to a node and its negation is called incoherent path pair. The pair of paths is minimal incoherent path pair (MIPP) if the two paths have no common arcs and each of them has no cycles.

Example 3 (Example 1 continued)**.** As shown in Fig. 2, the bold ellipses denote the aligned path-unsatisfiable nodes, and the bold arcs, either dotted or not, show the MIPPs for the aligned path-unsatisfiable nodes Reviewer_1 and Chair_1, i.e., mipp_a = [path1, path2] and mipp_b = [path3, path4], where
$\text{path1} = \text{Reviewer}_1 \rightarrow \text{Reviewer}_2 \rightarrow \text{Person}_2$,
$\text{path2} = \text{Reviewer}_1 \rightarrow \text{Review}_2 \rightarrow \text{Document}_2 \rightarrow \neg\text{Person}_2$,
$\text{path3} = \text{Chair}_1 \rightarrow \text{PC-Chair}_2 \rightarrow \text{ProgramCommitee}_1 \rightarrow \text{ConferenceMember}_1$,
$\text{path4} = \text{Chair}_1 \rightarrow \text{ConferenceChair}_2 \rightarrow \text{Conference}_1 \rightarrow \neg\text{ConferenceMember}_1$.

Two MIPPs mipp_a and mipp_b actually correspond to the two MIPSs shown in Example 2.

Note that not every aligned path-unsatisfiable node has a corresponding MIPP, whereas for each MIPP there is a unique aligned path-unsatisfiable node.

Corollary 1. *Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies and their mappings $\mathcal{M}$. $\mathcal{M}$ is incoherent iff there exists at least one MIPP in $\mathcal{G}$.*

The following theorem shows that the problem of computing the MIPSs of $O_i \cup O_j \cup \mathcal{M}$ can be reduced to finding all the MIPPs in the integrated graph $\mathcal{G}$.

Theorem 3. *Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i, O_j and their mappings $\mathcal{M}$. There exists a one-to-one correspondence between the set of MIPPs in $\mathcal{G}$ and the set of MIPSs of $O_i \cup O_j \cup \mathcal{M}$, in the way that every arc in MIPP corresponds to an inclusion or mapping axiom in MIPS and vice versa.*

Proof. ($\Rightarrow$) For every MIPP, let $\mathcal{M}_{\text{mipp}}$ be the set of inclusion and mapping axioms corresponding to its arcs. For any subset $\mathcal{M}' \subset \mathcal{M}_{\text{mipp}}$, we can construct a graph $\mathcal{G}_{\mathcal{M}'}$ applying the construction rules R1–R6 and MR1–MR6. Based on Definition 8, there does not exist any MIPPs in $\mathcal{G}_{\mathcal{M}'}$, thus $\mathcal{M}'$ is coherent. Therefore, $\mathcal{M}_{\text{mipp}}$ is a MIPS.

($\Leftarrow$) For every MIPS, we can get the set of arcs corresponding to its mappings and inclusions through applying the construction rules R1—R6 and MR1–MR6,

AUTHOR COPY

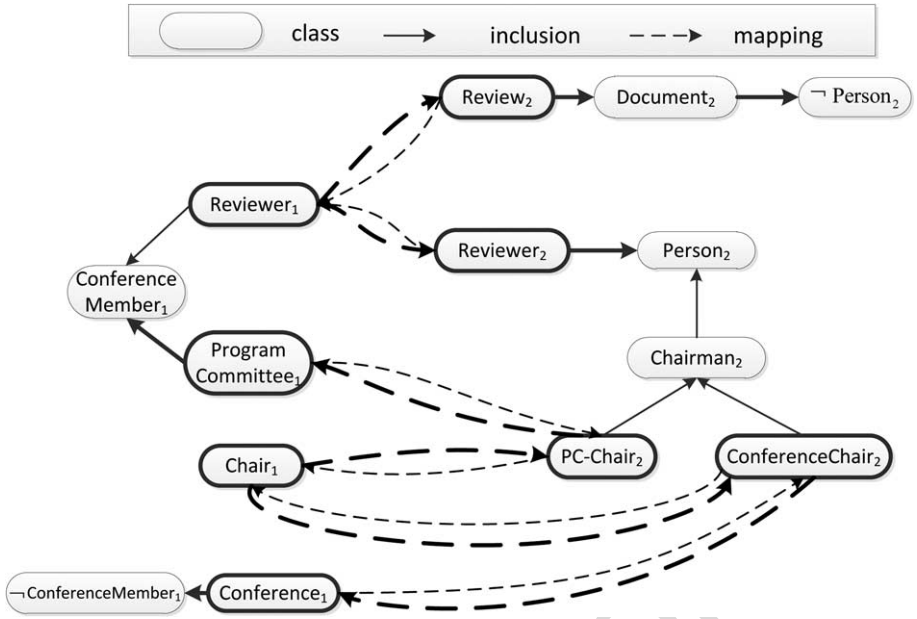

Fig. 2. Two MIPPs in the integrated graph according to Example 3.

denoted as $\mathcal{S}_{\text{mips}}$. According to Corollary 1, for any subset $\mathcal{S}' \subset \mathcal{S}_{\text{mips}}$, there exists a MIPP in $\mathcal{G}_{\mathcal{S}_{\text{mips}}}$ that does not occur in $\mathcal{G}_{\mathcal{S}'}$. Therefore, $\mathcal{S}_{\text{mips}}$ is a MIPP. □

5.2. *The graph-based algorithm for detecting incoherent mappings*

Our graph-based algorithm for detecting incoherence mappings is shown in Algorithm 1, which presents a general workflow for finding all the MIPSs based on the integrated graph. Concretely, rules R1–R6 and MR1–MR6 are applied to constructing an integrated graph first, and then Algorithm 2 is called to calculate the minimal incoherent path pairs (MIPPs), followed by Algorithm 3 designed to transform MIPPs into the format of inclusion axioms.

Although the workflow seems straightforward, Algorithm 2 is of high complexity because the task of obtaining minimal incoherent path pairs (MIPPs) in Step 6 is unfortunately in exponential time [14]. Nevertheless, Algorithm 2 only processes the pairs of disjoint nodes rather than all the nodes, which may alleviate the performance deficiency in practice. Moreover, MIPPs keep resourceful information about the incoherence including aligned path-unsatisfiable nodes and the relevant paths, which can be used to provide explanations to human about the provenance of incoherence.

Algorithm 1: A graph-based algorithm for detecting incoherent mappings.

Input: Two DL-Lite ontologies O_i and O_j, and mappings $\mathcal{M}$;
Output: The collection of MIPSs $\mathbb{M}$;
1 MIPPs $\longleftarrow \emptyset$, $\mathbb{M} \longleftarrow \emptyset$;
2 Construct $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ according to construction rules R1–R6 and MR1–MR6;
3 MIPPs $\longleftarrow$ CalculateMIPP($\mathcal{G}$);
4 **for** *each MIPP* $\in$ *MIPPs* **do**
5 $\quad \mathbb{M} \longleftarrow \mathbb{M} \cup$ TransformSoucreMappings($\mathcal{G}$,MIPP);
6 **return** $\mathbb{M}$;

Theorem 4. *Given two DL-Lite ontologies O_i and O_j and their mappings $\mathcal{M}$, Algorithm 1 returns the set of all the MIPSs of $O_i \cup O_j \cup \mathcal{M}$.*

Proof. According to Theorem 1, axioms of $O_i \cup O_j \cup \mathcal{M}$ are encoded completely in the integrated graph, thus all unsatisfiable concepts and roles related to MIPSs can be detected by traversing pairs of disjoint nodes. By Theorem 3, the output of Algorithm 1 corresponds to all MIPSs of $O_i \cup O_j \cup \mathcal{M}$. □

Example 4. (Example 1 continued). As shown in Fig. 2, for one pair of the disjoint nodes such as

AUTHOR COPY

Algorithm 2: CalculateMIPP($\mathcal{G}$).

Input: The constructed graph $\mathcal{G}$;
Output: The set of MIPPs in $\mathcal{G}$;
1 MIPPs $\longleftarrow \emptyset$;
2 **for** *each pair of nodes* S *and* $\neg S \in \mathcal{G}$ **do**
3 In $\mathcal{G}$, get the set of nodes on the paths to S, get the set of nodes on the paths to $\neg S$, and assign their intersection to set N_{unsat};
4 **if** N_{unsat} *is non-empty* **then**
5 **for** *each aligned path-unsatisfied node* $S_u \in N_{\text{unsat}}$ **do**
6 MIPP $\longleftarrow [\text{path}_{S_u \to \neg S}, \text{path}_{S_u \to S}]$;
7 MIPPs $\longleftarrow$ MIPP $\cup$ MIPPs;
8 **return** MIPPs;

Algorithm 3: TransformSoucreMappings.

Input: A set of mapping arcs *Arcs* in $\mathcal{G}$;
Output: A set of inclusions Axioms;
1 Axioms $\longleftarrow \emptyset$;
2 **for** *each arc* $\langle B_1, B_2 \rangle \in$ *Arcs* **do**
3 **if** B_1 *is in the form of* $\exists R_1$ *and* B_2 *the form of* $\exists R_2$ **then**
4 Axioms $\longleftarrow$ Axioms $\cup \{(R_1, R_2, \sqsubseteq)\}$;
5 **else**
6 Axioms $\longleftarrow$ Axioms $\cup \{(B_1, B_2, \sqsubseteq)\}$;
7 **return** Axioms;

Person_2, $\neg\text{Person}_2$, we first get the sets of nodes on the paths to Person_2 and $\neg\text{Person}_2$ and obtain the intersection of these two sets $\{\text{Reviewer}_1, \text{Review}_2, \text{Reviewer}_2\}$. Then, we identify the paths from the node in intersection to the nodes Person_2 and $\neg\text{Person}_2$ without cycle. Finally, three MIPPs of disjoint nodes Person_2 and $\neg\text{Person}_2$ are calculated. By Algorithm 2, eight MIPPs are identified as follows:
$\text{mipp}_a = [\text{Reviewer}_1 \to \text{Reviewer}_2 \to \text{Person}_2, \text{Reviewer}_1 \to \text{Review}_2 \to \text{Document}_2 \to \neg\text{Person}_2]$,
$\text{mipp}_b = [\text{Review}_2 \to \text{Reviewer}_1 \to \text{Reviewer}_2 \to \text{Person}_2, \text{Review}_2 \to \text{Document}_2 \to \neg\text{Person}_2]$,
$\text{mipp}_c = [\text{Reviewer}_2 \to \text{Person}_2, \text{Reviewer}_2 \to \text{Reviewer}_1 \to \text{Review}_2 \to \text{Document}_2 \to \neg\text{Person}_2]$,
$\text{mipp}_d = [\text{Chair}_1 \to \text{PC-Chair}_2 \to \text{ProgramCommitee}_1 \to \text{ConferenceMember}_1, \text{Chair}_1 \to \text{ConferenceChair}_2 \to \text{Conference}_1 \to \neg\text{ConferenceMember}_1]$,
$\text{mipp}_e = [\text{PC-Chair}_2 \to \text{ProgramCommittee}_1 \to \text{ConferenceMember}_1, \text{PC-Chair}_2 \to \text{Chair}_1 \to \text{ConferenceChair}_2 \to \text{Conference}_1 \to \neg\text{ConferenceMember}_1]$,
$\text{mipp}_f = [\text{ConferenceChair}_2 \to \text{Chair}_1 \to \text{PC-Chair}_2 \to \text{ProgramCommittee}_1 \to \text{ConferenceMember}_1, \text{ConferenceChair}_2 \to \text{Conference}_1 \to \neg\text{ConferenceMember}_1]$,
$\text{mipp}_g = [\text{ProgramCommitee}_1 \to \text{ConferenceMember}_1, \text{ProgramCommittee}_1 \to \text{PC-Chair}_2 \to \text{Chair}_1 \to \text{ConferenceChair}_2 \to \text{Conference}_1 \to \neg\text{ConferenceMember}_1]$,
$\text{mipp}_h = [\text{Conference}_1 \to \text{ConferenceChair}_2 \to \text{Chair}_1 \to \text{PC-Chair}_2 \to \text{ProgramCommittee}_1 \to \text{ConferenceMember}_1, \text{Conference}_1 \to \neg\text{ConferenceMember}_1]$.

6. Repairing incoherent mappings

To repair incoherent mappings, normally mappings with lower weights are removed. Such repairing strategy heavily relies on the weights assigned by individual ontology matchers and does not necessarily remove the wrong mappings. We propose three principles in this section as an attempt to identify and measure the wrong mappings, which features the notion of common closures of the integrated graph w.r.t. mapping arcs. First, the notion is introduced, followed by the three principles and their formalizing functions. To relieve the loss of information, we further define a mapping revision operator based on three functions. Lastly, we present the graph-based algorithm for repairing mappings.

6.1. The common closures with regard to mapping arcs

With Algorithms 1–3, we can obtain the collection of MIPSs with regard to ontologies and their mappings. Although removing any one mapping from each MIPS can solve the incoherence, wrong mappings may still remain in the alignment, as shown by the following example.

Example 5 (Example 4 continued)**.** In order to solve the conflicts in MIPSs, we need to remove one mapping from each MIPS. Assume that the removed map-

AUTHOR COPY

pings $\mathcal{M}_R$ and the left mappings $\mathcal{M}_L$ are as follows:

$$\begin{aligned}\mathcal{M}_R = \{&(\text{Reviewer}_1, \text{Review}_2, \sqsubseteq),\\ &(\text{Chair}_1, \text{PC-Chair}_2, \sqsubseteq),\\ &(\text{Review}_2, \text{Reviewer}_1, \sqsubseteq),\\ &(\text{PC-Chair}_2, \text{Chair}_1, \sqsubseteq)\}\\ \mathcal{M}_L = \{&(\text{Reviewer}_1, \text{Reviewer}_2, \sqsubseteq),\\ &(\text{Reviewer}_2, \text{Reviewer}_1, \sqsubseteq),\\ &(\text{Chair}_1, \text{ConferenceChair}_2, \sqsubseteq),\\ &(\text{ConferenceChair}_2, \text{Conference}_1, \sqsubseteq),\\ &(\text{ConferenceChair}_2, \text{Chair}_1, \sqsubseteq),\\ &(\text{Conference}_1, \text{ConferenceChair}_2, \sqsubseteq),\\ &(\text{PC-Chair}_2, \text{ProgramCommittee}_1, \sqsubseteq),\\ &(\text{ProgramCommittee}_1, \text{PC-Chair}_2, \sqsubseteq)\}\end{aligned}$$

Although the coherence is regained after removing $\mathcal{M}_R$, wrong mappings like (Conference$_1$, ConferenceChair$_2$, $\sqsubseteq$) and (ConferenceChair$_2$, Conference$_1$, $\sqsubseteq$) are still contained in the alignment.

In the integrated graph, mapping arcs build linkages between nodes across ontologies. We observe that such a linkage can be supported by multiple mapping arcs, or a singular mapping arc. We speculate that in terms of mappings in one MIPS, correct mappings are more likely to support the same linkages with other mappings across ontologies. This says that the linkages caused by singular mappings are less reliable, thus more probably being wrong. Next, we introduce the notion of common closures of the integrated graph w.r.t. mapping arcs to formalize such a speculation.

Definition 9 (Common Closures with regard to Mapping Arc)**.** In an integrated graph $\mathcal{G} = (N, E)$, the closures of $\mathcal{G}$ w.r.t. a mapping arc and a node S_1 are the arcs $\{\langle S_1, S_t\rangle\}$ in (N, E^*) satisfying that the path from S_1 to S_t passes this mapping arc. The common closures of $\mathcal{G}$ w.r.t. two mapping arcs from different paths in a MIPP is the intersection of closures w.r.t. these two mappings and the aligned path-unsatisfiable node corresponding to this MIPP, and the common closures of $\mathcal{G}$ w.r.t. one mapping arc are the union of such intersections w.r.t. it and other mapping arcs in all MIPPs.

We denote the common closures w.r.t. a mapping arc marc in a MIPP as CC(marc, mipp), and the common closures w.r.t. marc in all the MIPPs can be calculated by $\bigcup_{\text{mipp}\in\text{MIPPs}} \text{CC}(\text{marc}, \text{mipp})$, denoted as $\mathbb{CC}(\text{marc})$.

Example 6 (Example 4 continued)**.** Consider mapping arcs $\langle\text{Chair}_1, \text{PC-Chair}_2\rangle$, $\langle\text{Chair}_1, \text{ConferenceChair}_2\rangle$, $\langle\text{PC-Chair}_2, \text{ProgramCommittee}_1\rangle$, and $\langle\text{ConferenceChair}_2, \text{Conference}_1\rangle$ in mipp_d. As shown in Fig. 2, the closures w.r.t. them and aligned path-unsatisfiable node Chair_1 are listed as follows:
$\{\langle\text{Chair}_1, \text{Chairman}_2\rangle, \langle\text{Chair}_1, \text{Person}_2\rangle, \langle\text{Chair}_1, \text{ProgramCommittee}_1\rangle, \langle\text{Chair}_1, \text{ConferenceMember}_1\rangle\}$,
$\{\langle\text{Chair}_1, \text{Chairman}_2\rangle, \langle\text{Chair}_1, \text{Person}_2\rangle, \langle\text{Chair}_1, \text{Conference}_1\rangle, \langle\text{Chair}_1, \neg\text{ConferenceMember}_1\rangle\}$,
$\{\langle\text{Chair}_1, \text{ProgramCommittee}_1\rangle, \langle\text{Chair}_1, \text{ConferenceMember}_1\rangle\}$,
$\{\langle\text{Chair}_1, \text{Conference}_1\rangle, \langle\text{Chair}_1, \neg\text{ConferenceMember}_1\rangle\}$.

For mapping arc $\langle\text{Chair}_1, \text{PC-Chair}_2\rangle$, the common closures w.r.t. it and $\langle\text{Chair}_1, \text{ConferenceChair}_2\rangle$ is $\{\langle\text{Chair}_1, \text{Chairman}_2\rangle, \langle\text{Chair}_1, \text{Person}_2\rangle\}$, whereas the common closures w.r.t. it and other two mapping arcs are both empty sets. Therefore, $\text{CC}(\langle\text{Chair}_1, \text{PC-Chair}_2\rangle, \text{mipp}_d) = \{\langle\text{Chair}_1, \text{Chairman}_2\rangle, \langle\text{Chair}_1, \text{Person}_2\rangle\}$.

As the common closures w.r.t. it in other MIPPs are all empty sets, $\mathbb{CC}(\langle\text{Chair}_1, \text{PC-Chair}_2\rangle) = \{\langle\text{Chair}_1, \text{Chairman}_2\rangle, \langle\text{Chair}_1, \text{Person}_2\rangle\}$.

6.2. *The principles for removing mappings*

We propose three principles for repairing the incoherent mappings in the integrated graph as follows:

- P1: The number of the removed mapping arcs should be as small as possible.
- P2: The number of common closures with regard to the removed mapping arcs should be as small as possible.
- P3: The weights of mappings corresponding to the removed mapping arcs should be as low as possible.

Principle P1 is based on our observation that wrong mappings tend to cause more conflicts, thus occur in more MIPPs. We can count the occurrence of each mapping arc in all MIPPs and select the more frequent ones as candidates of removal. Principe P2 is based on our observation that wrong mapping arcs may relate to fewer common closures in MIPPs. Note that principles P1 and P2 are independent of individual match-

AUTHOR COPY

ers. Conversely, principle P3 relies on the weights of mappings generated by concrete matchers. The lower the weight, the less reliable the mapping is, thus more likely to be removed, a strategy adopted by many incoherence repair systems. Next, we formalize principles P1–P3 by three functions, respectively.

Definition 10 (Incision Function)**.** Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i, O_j and their mappings $\mathcal{M}$. $\mathbb{S}$ is the set of all the MIPPs in $\mathcal{G}$. An incision function for $\mathbb{S}$, denoted by σ, is a function ($\sigma : 2^{\mathbb{S}} \rightarrow 2^{E_{\mathcal{M}}}$) satisfying that

1. for each mipp $\in \mathbb{S}$, mipp $\cap\, \sigma(\mathbb{S}) \neq \emptyset$, and
2. the graph $\mathcal{G}' = (N, E \cup (E_{\mathcal{M}} \setminus \sigma(\mathbb{S})))$ is coherent.

The incision function selects at least one mapping arc in each MIPP to resolve the conflict. To meet the restriction of principle P1, the number of mapping arcs in $\sigma(\mathbb{S})$ should be minimum for the given $\mathbb{S}$.

Definition 11 (Closure Function)**.** Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i, O_j and their mappings $\mathcal{M}$. $\mathbb{S}$ is the set of all the MIPPs in $\mathcal{G}$. A closure function for $\sigma(\mathbb{S})$, denoted by τ, is a function ($\tau : 2^{E_{\mathcal{M}}} \rightarrow 2^{(E \cup E_{\mathcal{M}})^*}$) satisfying $\tau(\sigma(\mathbb{S})) = \bigcup_{\text{marc} \in \sigma(\mathbb{S})} \mathbb{CC}(\text{marc})$, where $\mathbb{CC}(\text{marc})$ is the common closures of $\mathcal{G}$ w.r.t. mapping arc marc in $\mathbb{S}$.

The closure function formalizes Principle P2 by counting the common closures of the removed mapping arcs in the integrated graph. To meet principle P2, the number of common closures of $\tau(\sigma(\mathbb{S}))$ should be minimum for $\sigma(\mathbb{S})$.

Definition 12 (Weight Function)**.** Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i, O_j and their mappings $\mathcal{M}$. $\mathbb{S}$ is the set of all the MIPPs in $\mathcal{G}$. A weight function for $\sigma(\mathbb{S})$, denoted by ω, is a function ($\omega : \sigma(\mathbb{S}) \rightarrow \mathbb{P}$) satisfying $\omega(\sigma(\mathbb{S})) = \sum_{\text{marc} \in \sigma(\mathbb{S})} n_{\text{marc}}$, where $\mathbb{P}$ represents positive rational numbers and n_{marc} is the weight of the mapping in $\mathcal{M}$ that corresponds to mapping arc marc.

The weight function sums up the weights of the removed mapping arcs. In order to satisfy Principle P3, the value of $\omega(\sigma(\mathbb{S}))$ should be minimum for $\sigma(\mathbb{S})$.

We attempt to use these functions to measure and identify the wrong mapping arcs in the integrated graph from different perspectives. To satisfy the conditions of P1–P3, we define a composite function by combining σ, τ and ω for the given $\mathbb{S}$, denoted as $\mathcal{F}$, for which there does not exist function $\mathcal{F}'$ combining σ', τ' and ω' satisfying the following conditions:

1. $|\sigma'(\mathbb{S})| < |\sigma(\mathbb{S})|$.
2. If $|\sigma'(\mathbb{S})| = |\sigma(\mathbb{S})|$, then $|\tau'(\sigma(\mathbb{S}))| < |\tau(\sigma(\mathbb{S}))|$.
3. If $|\sigma'(\mathbb{S})| = |\sigma(\mathbb{S})|$ and $|\tau'(\sigma(\mathbb{S}))| = |\tau(\sigma(\mathbb{S}))|$, then $\omega'(\sigma(\mathbb{S})) < \omega(\sigma(\mathbb{S}))$.

6.3. The mapping revision operator

In the process of repairing the inconsistency within ontology, implicit axioms entailed from the removed axioms are discarded, and in order to recover such loss of information, revision operators are defined to add these implied axioms back to the ontology as long as the inconsistency remains [11–14]. Similarly, we can define a mapping revision operator to relieve the loss of information entailed by the removed mappings under the condition that the coherence is not jeopardized. More specifically, we focus on adding back common closures of the integrated graph w.r.t. the removed mapping arcs.

Definition 13 (Mapping Revision Operator)**.** Let $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ be the integrated graph constructed from two DL-Lite ontologies O_i, O_j and their mappings $\mathcal{M}$. $\mathbb{S}$ is the set of all the MIPPs with regard to $\mathcal{G}$. A mapping revision operator with respect to $\mathcal{G}$ is a function $\upsilon : 2^{\mathbb{S}} \rightarrow 2^{(E \cup E_{\mathcal{M}})^*}$ satisfying that $\upsilon(\mathbb{S}) = (E_{\mathcal{M}} \setminus \mathcal{F}(\mathbb{S})) \cup \tau(\sigma(\mathbb{S}))$ and the graph $\mathcal{G}' = (N, E \cup \upsilon(\mathbb{S}))$ is coherent.

Example 7 (Example 6 continued)**.** Consider mapping arc $\langle \text{Chair}_1, \text{PC-Chair}_2 \rangle$ in mipp_d, the common closures w.r.t. it being $\{\langle \text{Chair}_1, \text{Chairman}_2 \rangle, \langle \text{Chair}_1, \text{Person}_2 \rangle\}$. When this mapping arc is removed, the common closures can be added back to the integrated graph as they do not cause any incoherence. This is the same case for mapping arc $\langle \text{Chair}_1, \text{ConferenceChair}_2 \rangle$.

Such added-back common closures w.r.t. mapping arcs are implied inclusions between concepts across ontologies, representing potential semantic bridges. To evaluate their reliability, weights of the mappings can be exploited. More concretely, the weight of a path can be computed by multiplying the weights of its arcs; and the weight of an element in the common closures is $(1 - (1 - \text{weight}_1) \times (1 - \text{weight}_2))$ where weight_1 and weight_2 are the weights of two paths in the corresponding MIPP. For instance, two common closures have the same weight in Example 7 which is

AUTHOR COPY

$1-(1-0.7)\times(1-0.5) = 0.85$. Common closures with higher weights are more likely to be potential mappings, and can be provided to human experts to make final decisions.

6.4. The graph-based algorithm for repairing incoherent mappings

Now, we present Algorithm 4 based on above formalized functions. Given graph $\mathcal{G}$ constructed from two input ontologies and their mappings, Steps 2–3 call Algorithm 2 to obtain the MIPPs and collect all the mapping arcs in MIPPs, denoted as *Arcs*. Steps 4–21 are a concrete realization of the mapping revision operator. Globally, it is intractable to calculate the closure function and the weight function, thus we adopt an approximation implementation: in each while loop, one mapping arc will be removed according to the functions until the conflicts of all the MIPPs are resolved. During this process, common closures of the removed arcs are saved as candidate arcs. Mapping revision operator implemented in Steps 4–21 can be computed in linear time in the size of *Arcs*. Nevertheless, as calculating MIPP takes exponential time [14], the complexity of Algorithm 4 is still in exponential time.

After removing mapping arcs and saving candidate arcs, Algorithm 5 is invoked to update the integrated graph. Steps 2–5 subsequently check whether the candidate arcs can be added into the graph, based on whether the coherence is kept. Finally, the updated mapping arcs are transformed into the mapping axioms by Algorithm 3.

Example 8 (Example 4 continued)**.** Algorithm 2 returns eight MIPPs, and their mapping arcs are listed as follows with weights:

$\langle \text{Reviewer}_1, \text{Review}_2, 0.8\rangle$

$\langle \text{Review}_2, \text{Reviewer}_1, 0.8\rangle$

$\langle \text{Reviewer}_1, \text{Reviewer}_2, 0.9\rangle$

$\langle \text{Reviewer}_2, \text{Reviewer}_1, 0.9\rangle$

$\langle \text{Chair}_1, \text{PC-Chair}_2, 0.5\rangle$

$\langle \text{PC-Chair}_2, \text{Chair}_1, 0.5\rangle$

$\langle \text{ConferenceChair}_2, \text{Conference}_1, 0.6\rangle$

$\langle \text{Conference}_1, \text{ConferenceChair}_2, 0.6\rangle$

$\langle \text{Chair}_1, \text{ConferenceChair}_2, 0.7\rangle$

$\langle \text{ConferenceChair}_2, \text{Chair}_1, 0.7\rangle$

Algorithm 4: Graph-based algorithm for mapping revision.

Input: The graph $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$ constructed from ontologies and incoherent mappings;
Output: Revised mappings $\mathcal{M}'$;

1 RemovedArcs ⟵ ∅, CandidateArcs ⟵ ∅;
2 MIPPs ⟵ CalculateMIPP($\mathcal{G}$);
3 *Arcs* ⟵ Collect the set of all the mapping arcs in MIPPs;
4 **while** *MIPPs is non-empty* **do**
5 N_{arc} ⟵ Count the number of each mapping arc of *Arcs* appearing in MIPPs;
6 RankedNumberArcs ⟵ SortNumber(*Arcs*, N_{arc});
7 **if** *Maximum number in RankedNumberArcs is unique* **then**
8 removedArc ⟵ arc having maximum number in RankedNumberArcs;
9 **else**
10 removedArcSet ⟵ arcs having maximum number in RankedNumberArcs;
11 RankedClosureArcs ⟵ SortCommonClosure(removedArcSet);
12 **if** *Minimum common closures in RankedClosureArcs is unique* **then**
13 removedArc ⟵ arc with minimum common closures in RankedClosureArcs;
14 **else**
15 removedArcSet ⟵ arcs with minimum common closures in RankedClosureArcs;
16 RankedWeightArcs ⟵ SortWeight(removedArcSet);
17 removedArc ⟵ Get the arc with minimum weight in RankedWeightArcs;
18 RemovedArcs ⟵ RemovedArcs ∪ removedArc;
19 CandidateArcs ⟵ CandidateArcs ∪ $\mathbb{CC}$(removedArc);
20 MIPPs ⟵ MIPPs\{MIPP | removedArc∈ MIPP};
21 *Arcs* ⟵ *Arcs*\removedArc;
22 UpdateGraph ($\mathcal{G}$, RemovedArcs, CandidateArcs);
23 $\mathcal{M}'$ ⟵ TransformSoucreMappings($E_{\mathcal{M}}$);
24 **return** $\mathcal{M}'$;

AUTHOR COPY

Table 1
The intermediate results of iterations in the while loop of Algorithm 4

Iteration	removedArc in removedArcSet	Principles applied	MIPPs unsolved
1	⟨**ConferenceChair**$_2$, **Conference**$_1$, **0.6**⟩	Principle 1	mipp$_a$, mipp$_b$
	⟨PC-Chair$_2$, ProgramCommittee$_1$, 0.8⟩	Principle 3	mipp$_b$, mipp$_h$
2	⟨**Reviewer**$_1$, **Review**$_2$, **0.8**⟩	Principle 1	mipp$_b$, mipp$_h$
	⟨Reviewer$_1$, Reviewer$_2$, 0.9⟩	Principle 3	
3	⟨**Conference**$_1$, **ConferenceChair**$_2$, **0.6**⟩		mipp$_b$
	⟨Review$_2$, Reviewer$_1$, 0.8⟩	Principle 1	
	⟨Reviewer$_1$, Reviewer$_2$, 0.9⟩	Principle 2	
	⟨ConferenceChair$_2$, Chair$_1$, 0.7⟩	Principle 3	
	⟨Chair$_1$, PC-Chair$_2$, 0.5⟩		
	⟨PC-Chair$_2$, ProgramCommittee$_1$, 0.8⟩		
4	⟨**Review**$_2$, **Reviewer**$_1$, **0.8**⟩	Principle 1	None
	⟨Reviewer$_1$, Reviewer$_2$, 0.9⟩	Principle 3	

Algorithm 5: UpdateGraph($\mathcal{G}$, RemovedArcs, CandidateArcs).

Input: The graph $\mathcal{G} = (N, E \cup E_{\mathcal{M}})$, RemovedArcs and CandidateArcs;
Output: Updated graph $\mathcal{G}' = (N, E \cup E'_{\mathcal{M}})$;

```
1 E_M ⟵ E_M \ RemovedArcs;
2 for each arc ∈ CandidateArcs do
3     E_M ⟵ E_M ∪ arc;
4     if CalculateMIPP(G) ≠ ∅ then
5         E_M ⟵ E_M \ arc;
```

⟨ProgramCommittee$_1$, PC-Chair$_2$, 0.8⟩

⟨PC-Chair$_2$, ProgramCommittee$_1$, 0.8⟩

Table 1 summarizes the intermediate result of every iteration in the while loop of Algorithm 4. The second column shows the initial set of arcs calculated in Step 6 and the final removed arc in Step 18. The third column shows the effective principles for selecting the wrong mapping arcs in MIPPs, and the last column lists the left MIPPs in each loop.

In Algorithm 5, we delete the removed arcs in the graph, and check whether the candidate arcs will cause incoherence. As there exist no common closures of the removed mapping arcs, the final mapping arcs in the graph are as follows:

⟨Reviewer$_1$, Reviewer$_2$, 0.9⟩

⟨Reviewer$_2$, Reviewer$_1$, 0.9⟩

⟨Chair$_1$, PC-Chair$_2$, 0.5⟩

⟨PC-Chair$_2$, Chair$_1$, 0.5⟩

⟨Chair$_1$, ConferenceChair$_1$, 0.7⟩

⟨ConferenceChair$_2$, Chair$_1$, 0.7⟩

⟨ProgramCommittee$_1$, PC-Chair$_2$, 0.8⟩

⟨PC-Chair$_2$, ProgramCommittee$_1$, 0.8⟩

By Algorithm 3, these mapping arcs are transformed into the format of the original mappings.

7. Evaluation and results

7.1. The implementation in the evaluation

We implemented both the detection and repair algorithms in our system. Several open source software packages are used, including OWLAPI,[1] a tool for managing OWL ontologies [34], and Neo4j,[2] a high-performance graph database by Neo Technology [16] for defining and querying graphs.

In the evaluation, we compare our approach with the state-of-the-art mapping validation algorithms [5,7], described as follows.

1. **Alcomo-Greedy** has been specifically developed for the purpose of debugging ontology alignments. A greedy strategy is adopted to sort mappings according to their weights and remove those that cause incoherence during an augmenting process [5].

[1] http://owlapi.sourceforge.net/
[2] https://neo4j.com/

AUTHOR COPY

Table 2
A summary of the repair systems compared in the evaluation

System	Alcomo-Greedy	Alcomo-Optimal	LogMap	Our approach
Incoherence Detection	Pellet Reasoner	Pellet Reasoner	Dowling-Gallier Algorithm	Graph-Based Algorithm
Repair Strategy	Sort mappings by their weights. Remove mappings greedily during the augmenting process	Find all the MIPSs without inclusions. Remove mappings globally by weights	Find the mappings causing unsatisfiable concepts and roles. Remove mappings locally by weights	Find all the MIPPs. Remove mappings globally by three principles
Complexity	Polynomial Time	Exponential Time	Polynomial Time	Exponential Time

Table 3
The statistics of the evaluated ontologies

Ontology	Language	♯ Concepts	♯ Roles	♯ PIs	♯ NIs
cmt (Conference)	$\mathcal{ALCIN}(\mathcal{D})$	30	59	33	27
Conference (Conference)	$\mathcal{ALCHIF}(\mathcal{D})$	60	64	71	14
confOf (Conference)	$\mathcal{SIN}(\mathcal{D})$	39	36	64	43
edas (Conference)	$\mathcal{ALCOIN}(\mathcal{D})$	104	50	92	407
mouse (Anatomy)	$\mathcal{ALE}$	2744	3	4493	0
human (Anatomy)	$\mathcal{S}$	3304	2	5423	17

2. **Alcomo-Optimal** applies an exhaustive search algorithm to find the global optimal diagnosis [5].
3. **LogMap** consists of a mapping discovery and a mapping repair component [7], and we only use the latter in the evaluation. Its repair strategy is to remove the minimum weighted mappings locally to regain the satisfiability of concepts.

These algorithms and our approach are summarized in Table 2.

All the experiments were performed on a desktop computer with Intel® Core™ i7-2600 (3.4 GHz) and 8 GB RAM in Java 1.8. Our system[3] can be downloaded together with the datasets and results.

7.2. The ontologies used in the evaluation

The ontologies to be mapped are from the Conference Track and Anatomy Track in OAEI[4] (Ontology Alignment Evaluation Initiative), an annual campaign for evaluating ontology matching systems that attracts many participants all over the world. Table 3 shows the statistics of these ontologies.

As shown in Table 3, the expressivity of the languages underpinning these ontologies is beyond DL-Lite, thus axioms that cannot be expressed by DL-Lite need to be approximated [16]. The approximations are described as follows.

1. Axioms in the form of $C \sqsubseteq \exists R.D$ are replaced by $C \sqsubseteq \exists R$.
2. Role R having $C_{\mathcal{D}}$ as domain and $C_{\mathcal{R}}$ as range is represented as $\exists R \sqsubseteq C_{\mathcal{D}}$ and $\exists R^{-} \sqsubseteq C_{\mathcal{R}}$.
3. Axioms containing constructor $\sqcup$ in the form of $A \sqsubseteq B \sqcup C$ are replaced by $A \sqsubseteq B$ and $A \sqsubseteq C$; and when B and C are disjoint, either $A \sqsubseteq B$ or $A \sqsubseteq C$ is chosen so as to prevent A from being unsatisfiable.
4. Axioms of enumeration and cardinality are ignored.

These approximations are sound [35], i.e., axioms hold in the resultant DL-Lite ontology can also be entailed in the original ontology. However, not all the axioms in the original ontology hold in the DL-Lite version. The approximations have relaxed the restrictions of axioms, thus our graph-based approach is not complete when detecting incoherent mappings across ontologies beyond DL-Lite. This contrasts with using OWL 2 reasoners such as Pellet [36] and Hermit [37], which can obtain the complete incoherence whereas the efficiency can be low. For approximating OWL 2 ontologies into DL-Lite, balancing between the efficiency and expressivity is of importance [35,38,39], and we leave this for our future work.

[3] https://github.com/liweizhuo001/GraphBasedOntologyTask
[4] http://oaei.ontologymatching.org/

AUTHOR COPY

Table 4
The incoherent alignments generated by the matchers

Num	Ontology Pair	Matcher	♯Mappings	♯Wrong Mappings	Coherent
1	cmt-Conference	GMap	16	5	No
2	cmt-confOf	StringEquiv	6	2	No
3	cmt-edas	GMap	11	3	No
4	Conference-confOf	GMap	16	4	No
5	Conference-edas	FCA-Map	12	3	No
6	confOf-edas	StringEquiv	17	7	No
7	confOf-edas	GMap	20	8	No
8	mouse-human	StringEquiv	947	4	No
9	mouse-human	GMap	1345	114	No
10	mouse-human	FCA-Map	1361	92	No

7.3. The alignments used in the evaluation

For incoherent mappings, we select alignments generated by StringEquiv, GMap [40] and FCA-Map [41]. StringEquiv serves as the baseline for measuring the performance of matchers. GMap is a probabilistic scheme for ontology matching that combines the sum-product network and the noisy-or model. FCA-Map uses the Formal Concept Analysis formalism to cluster the commonalities across ontologies. Applying these three matchers to aligning the six ontologies in Table 3, ten of the resultant alignments are incoherent, as listed in Table 4. These alignments are used to evaluate our repair system. OAEI provides reference alignments, so we can compute the precision, recall, and F-measure to measure the effectiveness of mapping validating systems. We assume that mappings not in the reference alignment are wrong mappings. Note that wrong mappings do not necessarily cause the incoherence of alignment, whereas the incoherence must be caused by wrong mappings.[5]

7.4. The evaluation results

In order to measure the repair performance, we compute the standard precision, recall and F-measure of the repaired alignments by every revision system, represented by P', R' and F', respectively, in the following tables in this section. These measures are compared with those of the original alignments, represented by P, R, and F, respectively. Moreover, the quality of repair can be revealed by comparing the number of removed mappings (denoted as $\sharp\mathcal{M}_R$) and the number of wrong mappings among them (denoted as $\sharp\mathcal{M}_W$). Given the repaired alignment $\mathcal{M}'$ and reference alignment $\mathcal{A}$, the formulas about P', R' and F' are defined as follows.

$$P' = \frac{|\mathcal{M}' \cap \mathcal{A}|}{|\mathcal{M}'|} \qquad R' = \frac{|\mathcal{M}' \cap \mathcal{A}|}{|\mathcal{A}|}$$

$$F' = \frac{2 \times P' \times R'}{P' + R'}$$

To test the efficiency, we ran all the repair systems ten times for every alignments in Conference Track and five times for Anatomy Track, and average runtime is taken as the revision time.

7.4.1. A comparison with other repair systems

Table 5 lists the repair result of the seven alignments across ontologies in the Conference track. Overall, all the systems are able to detect the incoherence and remove mappings to regain coherence except one case, that is our system failed to identify the incoherence in the cmt-confOf alignment generated by StringEquiv. It derives from the inadequacy of our approach in approximating ontologies beyond the expressivity of DL-Lite. Nevertheless, in terms of the F-measure and the percentage of the wrong mappings removed, our approach achieves the best results in four out of seven alignments; Alcomo-optimal, also adopting a global repair strategy, performs better in three alignments; and Alcomo-Greedy and LogMap are relatively weaker as they are based on local removal strategies. Moreover, in terms of the precision, our approach performs the best in five out of seven alignments, indicating that more wrong mappings have been removed.

[5] Such assumption does not always hold, as in the case of the OAEI Large Biomedical Ontologies track, where the reference alignments are extracted from the UMLS Metathesaurus, and for mappings cause incoherence, OAEI labels them as "Unknown", i.e., neither correct nor incorrect in the evaluation [42].

AUTHOR COPY

Table 5
The repair result for the alignments in the Conference Track

Num	Alignment	Repair System	$P'(P)$	$R'(R)$	$F'(F)$	$\sharp\mathcal{M}_R$	$\sharp\mathcal{M}_W$	Time
1	cmt Conference by GMap	Alcomo-Greedy	0.667 (0.688)	0.667 (0.733)	0.667 (0.710)	1	0	1 s
		Alcomo-Optimal	**0.733 (0.688)**	**0.733 (0.733)**	**0.733 (0.710)**	**1**	**1**	1 s
		LogMap Our	0.692 (0.688)	0.600 (0.733)	0.643 (0.710)	3	1	0.4 s
		Approach	**0.733 (0.688)**	**0.733 (0.733)**	**0.733 (0.710)**	**1**	**1**	13.4 s
2	cmt confOf by StringEquiv	Alcomo-Greedy	0.667 (0.667)	0.250 (0.250)	0.364 (0.364)	0	0	1 s
		Alcomo-Optimal	**0.800 (0.667)**	0.250 (0.250)	**0.381 (0.364)**	**1**	**1**	1 s
		LogMap	**0.800 (0.667)**	0.250 (0.250)	**0.381 (0.364)**	**1**	**1**	0.4 s
		Our Approach	0.667 (0.667)	0.250 (0.250)	0.364 (0.364)	0	0	10.6 s
3	cmt edas by GMap	Alcomo-Greedy	0.800 (0.727)	**0.615 (0.615)**	0.696 (0.667)	1	1	1 s
		Alcomo-Optimal	0.800 (0.727	**0.615 (0.615)**	0.696 (0.667)	1	1	1 s
		LogMap	**1.000 (0.727)**	**0.615 (0.615)**	**0.762 (0.667)**	**3**	**3**	0.5 s
		Our Approach	**1.000 (0.727)**	0.462 (0.615)	0.632 (0.667)	5	3	83.5 s
4	Conference confOf by GMap	Alcomo-Greedy	0.800 (0.750)	**0.800 (0.800)**	0.800 (0.774)	1	1	1 s
		Alcomo-Optimal	0.800 (0.750)	**0.800 (0.800)**	0.800 (0.774)	1	1	1 s
		LogMap	0.692 (0.750)	0.600 (0.800)	0.643 (0.774)	3	0	0.5 s
		Our Approach	**0.923 (0.750)**	**0.800 (0.800)**	**0.857 (0.774)**	**3**	**3**	15.4 s
5	Conference edas by FCA-Map	Alcomo-Greedy	**0.900 (0.750)**	**0.529 (0.529)**	**0.667 (0.621)**	**2**	**2**	1 s
		Alcomo-Optimal	**0.900 (0.750)**	**0.529 (0.529)**	**0.667 (0.621)**	**2**	**2**	1 s
		LogMap	0.889 (0.750)	0.471 (0.529)	0.615 (0.621)	3	2	0.5 s
		Our Approach	0.875 (0.750)	0.412 (0.529)	0.560 (0.621)	5	2	19.5 s
6	confOf edas by StringEquiv	Alcomo-Greedy	0.625 (0.588)	**0.526 (0.526)**	0.571 (0.556)	1	1	1 s
		Alcomo-Optimal	0.667 (0.588)	**0.526 (0.526)**	0.588 (0.556)	2	2	3 s
		LogMap	0.692 (0.588)	0.474 (0.526)	0.562 (0.556)	4	3	0.5 s
		Our Approach	**0.818 (0.588)**	0.474 (0.526)	**0.600 (0.556)**	**6**	**5**	46.8 s
7	confOf edas by GMap	Alcomo-Greedy	0.611 (0.600)	**0.579 (0.632)**	0.595 (0.615)	2	1	1 s
		Alcomo-Optimal	0.647 (0.600)	**0.579 (0.632)**	0.611 (0.615)	3	2	3 s
		LogMap	0.750 (0.600)	0.474 (0.632)	0.581 (0.615)	7	4	0.4 s
		Our Approach	**0.769 (0.600)**	0.526 (0.632)	**0.625 (0.615)**	**7**	**5**	138.5 s

Table 6
The repair result for the alignments in the Anatomy Track

Num	Alignment	Repair System	$P'(P)$	$R'(R)$	$F'(F)$	$\sharp\mathcal{M}_R$	$\sharp\mathcal{M}_W$	Time
8	mouse human by StringEquiv	Alcomo-Greedy	0.997 (0.996)	**0.622 (0.622)**	**0.766 (0.766)**	**1**	**1**	4 s
		Alcomo-Optimal	–	–	–	–	–	–
		LogMap	0.997 (0.996)	0.621 (0.622)	0.765 (0.766)	3	1	4.2 s
		Our Approach	0.997 (0.996)	**0.622 (0.622)**	**0.766 (0.766)**	**1**	**1**	492 s
9	mouse human by GMap	Alcomo-Greedy	0.915 (0.915)	0.805 (0.812)	0.857 (0.861)	10	0	5 s
		Alcomo-Optimal	–	–	–	–	–	–
		LogMap	**0.917 (0.915)**	0.811 (0.812)	0.861 (0.861)	3	2	5.1 s
		Our Approach	0.916 (0.915)	**0.812 (0.812)**	**0.861 (0.861)**	**1**	**1**	1341 s
10	mouse human by FCA-Map	Alcomo-Greedy	0.951 (0.932)	0.814 (0.837)	0.877 (0.882)	64	29	5 s
		Alcomo-Optimal	–	–	–	–	–	–
		LogMap	0.935 (0.932)	0.832 (0.837)	0.881 (0.882)	11	4	4.5 s
		Our Approach	**0.935 (0.932)**	**0.836 (0.837)**	**0.883 (0.882)**	**6**	**4**	2451 s

Table 6 shows the repair result of the three alignments across ontologies in the Anatomy track. The optimality of Alcomo-Optimal is guaranteed via an exhaustive search algorithm to check potential solutions. This makes the system incompetent for large-scale mapping validation problems [9], as shown by all the tasks in Table 6. For Alcomo-Greedy, LogMap and our approach, the incoherence of each alignment is detected and resolved after repair. In terms of the F-measure and the percentage of the wrong mappings

AUTHOR COPY

Table 7
The contribution of repair principles

Num	Alignment	Principles	$P'(P)$	$R'(R)$	$F'(F)$	$\sharp\mathcal{M}_R$	$\sharp\mathcal{M}_W$
4	Conference confOf by GMap	P1	0.846 (0.750)	0.733 (0.800)	0.786 (0.774)	3	2
		P2	0.909 (0.750)	0.667 (0.800)	0.769 (0.774)	5	3
		P3	0.818 (0.750)	0.600 (0.800)	0.692 (0.774)	5	2
		P1 + P2	0.923 (0.750)	0.800 (0.800)	0.857 (0.744)	3	3
		P1 + P3	0.917 (0.750)	0.733 (0.800)	0.815 (0.774)	4	3
		P2 + P3	0.857 (0.750)	0.800 (0.800)	0.828 (0.774)	2	2
		P1 + P2 + P3	**0.923 (0.750)**	**0.800 (0.800)**	**0.857 (0.774)**	**3**	**3**
6	confOf edas by StringEquiv	P1	0.818 (0.588)	0.474 (0.526)	0.600 (0.566)	6	5
		P2	0.889 (0.588)	0.421 (0.526)	0.571 (0.556)	8	6
		P3	0.889 (0.588)	0.421 (0.526)	0.571 (0.556)	8	6
		P1 + P2	0.818 (0.588)	0.474 (0.526)	0.600 (0.566)	6	5
		P1 + P3	0.818 (0.588)	0.474 (0.526)	0.600 (0.556)	6	5
		P2 + P3	0.889 (0.588)	0.421 (0.526)	0.571 (0.556)	8	6
		P1 + P2 + P3	**0.818 (0.588)**	**0.474 (0.526)**	**0.600 (0.556)**	**6**	**5**
7	confOf edas by GMap	P1	0.769 (0.600)	0.526 (0.632)	0.625 (0.615)	7	5
		P2	0.800 (0.600)	0.421 (0.632)	0.552 (0.615)	10	6
		P3	0.750 (0.600)	0.474 (0.632)	0.581 (0.615)	8	5
		P1 + P2	0.769 (0.600)	0.526 (0.632)	0.625 (0.615)	7	5
		P1 + P3	0.769 (0.600)	0.526 (0.632)	0.625 (0.615)	7	5
		P2 + P3	0.750 (0.600)	0.474 (0.632)	0.581 (0.615)	8	5
		P1 + P2 + P3	**0.769 (0.600)**	**0.526 (0.632)**	**0.625 (0.615)**	**7**	**5**
10	mouse human by FCA-Map	P1	0.936 (0.932)	0.835 (0.837)	0.883 (0.882)	8	5
		P2	0.935 (0.932)	0.830 (0.837)	0.879 (0.882)	16	5
		P3	0.934 (0.932)	0.824 (0.837)	0.876 (0.882)	24	4
		P1 + P2	0.935 (0.932)	0.836 (0.837)	0.883 (0.882)	6	4
		P1 + P3	0.936 (0.932)	0.835 (0.837)	0.883 (0.882)	8	5
		P2 + P3	0.935 (0.932)	0.834 (0.837)	0.881 (0.882)	9	5
		P1 + P2 + P3	**0.935 (0.932)**	**0.836 (0.837)**	**0.883 (0.882)**	**6**	**4**

removed, our approach achieves the best result in all three alignments. Compared with the scalability problem of Alcomo-Optimal and local removal strategy of LogMap, our approach pursues a balance between the computational complexity of mapping validation and expressivity of aligned ontologies.

Alcomo-Greedy, Alcomo-Optimal and Logmap decide the mappings to be removed based on their weights. Solely relying on the weights in repair can be unreliable. More importantly, it is not suitable for alignments generated by qualitative matchers such as StringEquiv and FCA-Map which do not assign numerical weights to mappings. Our approach, on the other hand, combines three principles, where two are independent of the weights from specific matchers. Of note, not all wrong mappings in the alignment can cause the incoherence, as exemplified by the mouse and human anatomy alignment generated by GMap, where 114 mappings are wrong, and our approach solely removed one of them and then the coherence is regained.

In terms of the time consumption, our approach takes much longer times than the other systems, where a large proportion is spent on calculating MIPPs during the incoherence detection. Similarly to find MIPSs, computing MIPPs in graph is a NP-hard problem [14]. The optimization technologies like [43,44] may reduce the repeated calculations for our graph-based incoherence repair algorithm, which are worth exploring in our future work.

7.4.2. The contribution of each repair principle

In order to evaluate the contribution of each repair principle separately, we select four of the alignments that our approach performs better, as listed in Table 7. One can see that principle P1, a global principle is independent of specific matchers, serves as a good guidance for removing wrong mappings. Nevertheless, the order of removing with P1 is indecisive, and this is when principles P2 and P3 can complement to further improve the repair performance. For example, common closures can play a part in alignment 4 and 10, so combining P2 with P1 increased the F-measure and

AUTHOR COPY

Table 8
Applying repair principles in different orders

Num	Alignment	Principles	$P'(P)$	$R'(R)$	$F'(F)$	$\sharp\mathcal{M}_R$	$\sharp\mathcal{M}_W$
4	Conference	P1→ P2 →P3	**0.923 (0.750)**	**0.800 (0.800)**	**0.857 (0.774)**	**3**	**3**
	confOf by	P1→ P3 →P2	0.917 (0.750)	0.733 (0.800)	0.815 (0.774)	4	3
	GMap	P2→ P1 →P3	0.917 (0.750)	0.733 (0.800)	0.815 (0.774)	4	3
		P2→ P3 →P1	0.857 (0.750)	0.800 (0.800)	0.828 (0.744)	2	2
		P3→ P1 →P2	0.846 (0.750)	0.733 (0.800)	0.786 (0.774)	3	2
		P3→ P2 →P1	0.846 (0.750)	0.733 (0.800)	0.786 (0.774)	3	2

the proportion of the removed wrong mappings. When mapping arcs in MIPPs have no common closures, the improvements from P2 becomes limited. As matter of fact, P2 is more suitable for ontologies with rich taxonomic structures. Lastly, utilizing the three principles as a whole always achieved the best performance.

7.4.3. The order of applying the repair principles

We use the alignment between Conference and confOf generated by GMap to explore the influence of following different orders to apply the principles, as shown in Table 8. One can see that the composition of three principles in any order is better than applying singular principle. Moreover, considering principle P1 first would obtain a better repair, whereas the ordering of P2 and P3 following P1 is not certain. When the weights of mappings are reliable, it is recommended to apply P3 before P2, otherwise P2 can come before P3.

7.4.4. The common closures exploited by the mapping revision operator

In order to relieve the loss of information during the repair process, common closures of the removed mapping arcs are saved, and later added back to the integrated graph once determined that no further incoherence can be caused. In our evaluation, the number of common closures actually added back is limited. For example, only one common closure (mouse:organ system, human:Anatomic_Structure_System_or_Substance, $\sqsubseteq$, 1.0) is used after repairing the alignment between mouse and human anatomy generated by FCA-Map. Such common closures can be used by domain experts, for instance, to identify potential, new mappings across ontologies.

8. Conclusions

In this paper, we present a graph-based approach for resolving incoherent mappings among DL-Lite ontologies. We first encode ontologies and mappings into a graph according to the designed construction rules, in which the problem of computing the set of all the MIPSs can be reduced to obtaining the MIPPs based on the graph. In order to remove the wrong mappings in the repair stage, we propose three elaborated repair principles, and define functions that fulfill these principles, in which we introduce the notion of common closures of the integrated graph w.r.t. a mapping arc. To relieve the loss of information, we further design a mapping revision operator based on these functions, so that the common closures related to the removed mapping arcs can be added back to the graph when no incoherence occurs. Compared with some state-of-the-art systems on real-world ontologies, our approach can remove more wrong mappings and achieve better repairing results in most of the cases.

There are several avenues for our future work. Our graph-based mapping validation approach targets the family of DL-Lite ontologies, and in order to extend to more expressive description logics, we will focus on approximating OWL 2 ontologies into DL-Lite tailed for our algorithms. Recently, Solimando et al. presented an approach to detecting and minimizing violations in ontology alignments based on the conservativity principle, which is useful for measuring wrong mappings and negotiating about mappings in dialogues with humans [45,46]. In addition to knowledge base repairing based on logical incoherence, there are works modeling wrong relations that cannot be identified by logical reasoners. These works include data-driven methods based on top level ontologies and cycle methods based on is-a relations for improving the quality of the automatically constructed web-scale knowledge bases [47,48]. Incorporating such ideas into our method can facilitate discarding more wrong mappings. Importantly, optimization technologies shall be explored so as to improve the efficiency of our system. Lastly, our repair techniques will be extended in order to cater for other tasks and scenarios such as revision of complex ontology mappings and networks of ontologies [49,50].

AUTHOR COPY

Acknowledgements

This work has been supported by the National Key Research and Development Program of China under grant 2016YFB1000902, the Natural Science Foundation of China grants 61232015, 61621003, 61272378, supported in part by the 863 Program under Grant 2015AA015406, the Knowledge Innovation Program of the Chinese Academy of Sciences (CAS), and Institute of Computing Technology of CAS.

References

[1] R. Tolksdorf, L. Nixon and E. Simperl, Towards a tuplespace-based middleware for the semantic web, *Web Intelligence and Agent Systems: An International Journal* **6**(3) (2008), 235–251. doi:10.3233/WIA-2008-0139.

[2] L. Otero-Cerdeira, F.J. Rodríguez-Martínez and A. Gómez-Rodríguez, Ontology matching: A literature review, *Expert Systems with Applications* **42**(2) (2015), 949–971. doi:10.1016/j.eswa.2014.08.032.

[3] G. Qi, Q. Ji and P. Haase, A conflict-based operator for mapping revision, in: *Proceedings of the 8th International Semantic Web Conference*, Springer, 2009, pp. 521–536.

[4] M. Ruta, T. Di Noia, E. Di Sciascio and F.M. Donini, Semantic based collaborative P2P in ubiquitous computing, *Web Intelligence and Agent Systems: An International Journal* **5**(4) (2007), 375–391.

[5] C. Meilicke, Alignment incoherence in ontology matching, PhD thesis, Universitätsbibliothek Mannheim, 2011.

[6] J. Euzenat and P. Shvaiko, *Ontology Matching*, Springer Science & Business Media, 2013. doi:10.1007/978-3-642-38721-0.

[7] E. Jiménez-Ruiz and B.C. Grau, Logmap: Logic-based and scalable ontology matching, in: *Proceedings of the 10th International Semantic Web Conference*, Springer, 2011, pp. 273–288.

[8] S. Castano, A. Ferrara, D. Lorusso, T.H. Näth and R. Möller, Mapping validation by probabilistic reasoning, in: *Proceedings of the 5th European Semantic Web Conference*, Springer, 2008, pp. 170–184.

[9] J. Noessner, H. Stuckenschmidt, C. Meilicke and M. Niepert, Completeness and optimality in ontology alignment debugging, in: *Proceedings of the 9th International Conference on Ontology Matching*, CEUR-WS.org, 2014, pp. 25–36.

[10] P. Klinov, Practical reasoning in probabilistic description logic, PhD thesis, University of Manchester, 2011.

[11] G. Qi, P. Haase, Z. Huang, Q. Ji, J.Z. Pan and J. Völker, A kernel revision operator for terminologies – algorithms and evaluation, in: *Proceedings of the 7th International Semantic Web Conference*, Springer, 2008, pp. 419–434.

[12] J. Du, G. Qi and X. Fu, A practical fine-grained approach to resolving incoherent OWL 2 DL terminologies, in: *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*, ACM, 2014, pp. 919–928.

[13] G. Qi, Z. Wang, K. Wang, X. Fu and Z. Zhuang, Approximating model-based ABox revision in DL-lite: Theory and practice, in: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI Press, 2015, pp. 254–260.

[14] X. Fu, G. Qi, Y. Zhang and Z. Zhou, Graph-based approaches to debugging and revision of terminologies in DL-Lite, *Knowledge-Based Systems* **100** (2016), 1–12. doi:10.1016/j.knosys.2016.01.039.

[15] L. Zhou, H. Huang, G. Qi, Y. Ma, Z. Huang and Y. Qu, Paraconsistent query answering over DL-Lite ontologies, *Web Intelligence and Agent Systems* **10** (2012), 19–31. doi:10.3233/WIA-2012-0228.

[16] D. Lembo, V. Santarelli and D.F. Savo, Graph-based ontology classification in OWL 2 QL, in: *Proceedings of the 10th Extended Semantic Web Conference*, Springer, 2013, pp. 320–334.

[17] C. Meilicke and H. Stuckenschmidt, Applying logical constraints to ontology matching, in: *Proceedings of the 30th Annual Conference on Artificial Intelligence*, Springer, 2007, pp. 99–113.

[18] C. Meilicke, H. Stuckenschmidt and A. Tamilin, Repairing ontology mappings, in: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, AAAI Press, 2007, pp. 1408–1413.

[19] C. Meilicke, J. Völker and H. Stuckenschmidt, Learning disjointness for debugging mappings between lightweight ontologies, in: *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management*, Springer, 2008, pp. 93–108.

[20] Y.R. Jean-Mary, E.P. Shironoshita and M.R. Kabuka, Ontology matching with semantic verification, *Web Semantics: Science, Services and Agents on the World Wide Web* **7**(3) (2009), 235–251. doi:10.1016/j.websem.2009.04.001.

[21] M.G. Scutella, A note on Dowling and Gallier's top-down algorithm for propositional Horn satisfiability, *The Journal of Logic Programming* **8**(3) (1990), 265–273. doi:10.1016/0743-1066(90)90026-2.

[22] E. Santos, D. Faria, C. Pesquita and F.M. Couto, Ontology alignment repair through modularization and confidence-based heuristics, *PloS one* **10**(12) (2015), 1–19.

[23] T.H. Näth and R. Möller, ContraBovemRufum: A system for probabilistic lexicographic entailment, in: *Proceedings of the 21st International Workshop on Description Logics*, CEUR-WS.org, 2008.

[24] T. Lukasiewicz, Expressive probabilistic, *Artificial Intelligence* **172**(6) (2008), 852–883. doi:10.1016/j.artint.2007.10.017.

[25] P. Shvaiko and J. Euzenat, Ontology matching: State of the art and future challenges, *IEEE Transactions on Knowledge and Data Engineering* **25**(1) (2013), 158–176. doi:10.1109/TKDE.2011.253.

[26] T. Lukasiewicz and U. Straccia, Managing uncertainty and vagueness in description logics for the semantic web, *Web Semantics: Science, Services and Agents on the World Wide Web* **6**(4) (2008), 291–308. doi:10.1016/j.websem.2008.04.001.

[27] F. Riguzzi, E. Bellodi, E. Lamma and R. Zese, Probabilistic description logics under the distribution semantics, *Semantic Web* **6**(5) (2015), 477–501. doi:10.3233/SW-140154.

[28] T. Lukasiewicz, Probabilistic description logic programs, *International Journal of Approximate Reasoning* **45**(2) (2007), 288–307. doi:10.1016/j.ijar.2006.06.012.

AUTHOR COPY

[29] G. Qi, Q. Ji, J.Z. Pan and J. Du, Extending description logics with uncertainty reasoning in possibilistic logic, *International Journal of Intelligent Systems* **26**(4) (2011), 353–381. doi:10.1002/int.20470.

[30] T. Lukasiewicz, L. Predoiu and H. Stuckenschmidt, Tightly integrated probabilistic description logic programs for representing ontology mappings, *Annals of Mathematics and Artificial Intelligence* **63**(3) (2011), 385–425. doi:10.1007/s10472-012-9280-3.

[31] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini and R. Rosati, DL-Lite: Tractable description logics for ontologies, in: *Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference*, AAAI Press, 2005, pp. 602–607.

[32] F. Baader, *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2007. doi:10.1017/CBO9780511711787.

[33] S. Even, *Graph Algorithms*, Cambridge University Press, 2011. doi:10.1017/CBO9781139015165.

[34] M. Horridge and S. Bechhofer, The owl api: A Java api for owl ontologies, *Semantic Web* **2**(1) (2011), 11–21. doi:10.3233/SW-2011-0025.

[35] E. Botoeva, D. Calvanese and M. Rodriguez-Muro, Expressive approximations in DL-Lite ontologies, in: *Proceedings of the 13th International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, Springer, 2010, pp. 21–31. doi:10.1007/978-3-642-15431-7_3.

[36] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur and Y. Katz, Pellet: A practical owl-dl reasoner, *Web Semantics: Science, Services and Agents on the World Wide Web* **5**(2) (2007), 51–53. doi:10.1016/j.websem.2007.03.004.

[37] R. Shearer, B. Motik and I. Horrocks, HermiT: A highly-efficient OWL reasoner, in: *Proceedings of the 5th Workshop on OWL: Experiences and Directions*, CEUR-WS.org, 2008, pp. 91–100.

[38] M. Console, V. Santarelli and D.F. Savo, Efficient approximation in DL-Lite of OWL 2 ontologies, in: *Proceedings of the 26th International Workshop on Description Logics*, CEUR-WS.org, 2013, pp. 132–143.

[39] I. Horrocks, P.F. Patel-Schneider and F. Van Harmelen, From SHIQ and RDF to OWL: The making of a web ontology language, *Web Semantics: Science, services and agents on the World Wide Web* **1**(1) (2003), 7–26. doi:10.1016/j.websem.2003.07.001.

[40] W. Li and Q. Sun, GMap: Results for OAEI 2015, in: *Proceedings of the 10th International Workshop on Ontology Matching*, CEUR-WS.org, 2015, pp. 150–157.

[41] M. Zhao and S. Zhang, Identifying and validating ontology mappings by formal concept analysis, in: *Proceedings of the 11th International Workshop on Ontology Matching*, CEUR-WS.org, 2016, pp. 61–72.

[42] C. Pesquita, D. Faria, E. Santos and F.M. Couto, To repair or not to repair: Reconciling correctness and coherence in ontology reference alignments, in: *Proceedings of the 8th International Workshop on Ontology Matching*, CEUR-WS.org, 2013, pp. 13–24.

[43] A. Fijany and F. Vatan, New approaches for efficient solution of hitting set problem, in: *Proceedings of the Winter International Symposium on Information and Communication Technologies*, Trinity College Dublin, 2004.

[44] L. Lin and Y. Jiang, The computation of hitting sets: Review and new algorithms, *Information Processing Letters* **86**(4) (2003), 177–184. doi:10.1016/S0020-0190(02)00506-9.

[45] A. Solimando, E. Jiménez-Ruiz and G. Guerrini, Minimizing conservativity violations in ontology alignments: Algorithms and evaluation, *Knowledge and Information Systems* **51** (2016), 775–819. doi:10.1007/s10115-016-0983-3.

[46] E. Jiménez-Ruiz, T.R. Payne, A. Solimando and V.A.M. Tamma, Limiting logical violations in ontology alignment through negotiation, in: *Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning*, AAAI Press, 2016, pp. 217–226.

[47] H. Paulheim, Data-driven joint debugging of the DBpedia mappings and ontology – towards addressing the causes instead of the symptoms of data quality in DBpedia, in: *Proceedings of the 14th Extended Semantic Web Conference*, Springer, 2017, pp. 404–418.

[48] J. Liang, Y. Xiao, Y. Zhang, S.-w. Hwang and H. Wang, Graph-based wrong IsA relation detection in a large-scale lexical taxonomy, in: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI Press, 2017, pp. 1178–1184.

[49] N. Silva and J. Rocha, Semantic web complex ontology mapping, *Web Intelligence and Agent Systems: An International Journal* **1**(3) (2003), 235–248.

[50] J. Euzenat, Revision in networks of ontologies, *Artificial intelligence* **228** (2015), 195–216. doi:10.1016/j.artint.2015.07.007.

AUTHOR COPY