UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

CAUÃ ROCA ANTUNES

Detecting and correcting conservativity violations in ontology alignments under a category-theoretic framework

Dissertation presented in partial fulfilment of the requirements for the degree of Master in Computer Science.

Advisor: Prof. Dr. Mara Abel
Co-advisor: Prof. Dr. Alexandre Rademaker

Porto Alegre
April 2020

CIP – CATALOGING-IN-PUBLICATION

Roca Antunes, Cauã
Detecting and correcting conservativity violations in ontology alignments under a category-theoretic framework / Cauã Roca Antunes. – 2020.
58 f.:il.
Advisor: Mara Abel; Co-advisor: Alexandre Rademaker.
Dissertation (Masters) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2020.
1. Ontology alignment. 2. Category theory. 3. Semantic interoperability. I. Abel, Mara. II. Rademaker, Alexandre. III. Detecting and correcting conservativity violations in ontology alignments under a category-theoretic framework.

ABSTRACT

This work formalizes conservative and non-conservative ontology alignments in category theory, a branch of mathematics that studies systems of composable mappings, and proposes an algorithm for correcting conservativity violations in such alignments. Ontologies are computational artifacts that model the structure of portions of reality. When multiple ontologies dealing with related domains are to be used together, which is frequent in distributed contexts such as the Semantic Web, it is necessary to build an alignment between them, i.e., a set of mappings between entities in both ontologies. Such alignments frequently produce inconsistencies, even when constructed by domain experts. One kind of inconsistency which is often introduced is the violation of the conservativity principle, that states that merging two ontologies trough an alignment should not introduce new subsumption relations between concepts originating from the same source ontology. Alignments that do not violate the conservativity principle are called conservative alignments. This work presents a category-theoretic formalization of conservativity in ontology alignments, allowing the analysis of conservativity under the operations of alignment intersection, union and composition. Finally, the category-theoretic formalization provides the basis for a quadratic-time algorithm for correcting conservativity violations in ontology alignments. We evaluate the algorithm against datasets from the Ontology Alignment Evaluation Initiative 2019 and compare the results with the available state-of-the-art approach. Our algorithm was over 10 times faster for all datasets and performed less modifications over the original alignments, preserving more information from the input.

Keywords: Ontology alignment. Category theory. Semantic interoperability.

Detectando e corrigindo violações de conservatividade em alinhamentos de ontologias sob um arcabouço em teoria das categorias

RESUMO

Este trabalho formaliza alinhamentos de ontologias conservativos e não conservativos em teoria das categorias, um ramo da matemática que estuda sistemas de mapeamentos componíveis, e propõe um algoritimo para corrigir violações de conservatividade em tais alinhamentos. Ontologias são artefatos computacionais que modelam a estrutura de porções da realidade. Quando multiplas ontologias lidando com domínios relacionados são utilizadas conjuntamente, o que é frequente em contextos distribuídos tais como a Web Semântica, é necessária a construção de um alinhamento entre elas, isto é, de um conjunto de mapeamentos entre entidades de ambas ontologias. Tais alinhamentos frequentemente produzem inconsistências, mesmo quando produzidos por especialistas de domínio. Um tipo comum de inconsistência é a violação do princípio da conservatividade, que afirma que a fusão de duas ontologias a partir de um alinhamento não deveria introduzir novas relações de subsunção entre entidades vindas da mesma ontologia de origem. Alinhamentos que não violam o princípio de conservatividade são chamados de alinhamentos conservativos. Este trabalho apresenta uma formalização em teoria das categorias de conservatividade em alinhamentos de ontologias, permitindo a análise da conservatividade sob as operações de intersecção, união e composição de alinhamentos. Finalmente, a formalização em teoria das categorias fornece a base para um algoritmo de tempo quadrático para a correção de violações de conservatividade em alinhamentos de ontologia. O algoritmo foi avaliado com conjuntos de dados da Ontology Alignment Evaluation Initiative 2019, e os resultados foram comparados com uma abordagem do estado da arte. Nosso algoritmo foi mais de 10 vezes mais rápido para todos os casos de teste e operou um número significativamente menor de modificações sobre os alinhamentos originais, preservando mais informação da entrada.

Palavras-chave: Alinhamento de ontologias. Teoria das categorias. Interoperabilidade semântica.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABREVIATIONS AND ACRONYMS

OAEI	Ontology Alignment Evaluation Initiative
OWL	Web Ontology Language
UFRGS	Universidade Federal do Rio Grande do Sul

CONTENTS

1 INTRODUCTION

Unlike the traditional World Wide Web, which is intended for human access and usage, the Semantic Web is meant to be accessible by computer applications. To enable this, the meaning of data must be specified formally in ontologies, computational artifacts that explicitly describe some portion of reality according to the view of some community. Thus, ontologies specify the meaning of the concepts, while the instances of such concepts are described in an assertion component of the knowledge base.

Often, multiple communities produce different ontologies dealing with domains that are related or the same. In such cases, it is useful to combine the knowledge from those distinct ontologies. This integration is particularly important in distributed contexts such as the Semantic Web, where information may come from many different sources.

In order to integrate two ontologies, it is necessary to find mappings between concepts in both ontologies. The task of finding such mappings is called ontology matching, and the final result of this process is an ontology alignment.

Even though there are many approaches to match ontologies, both manually and automatically, none is free of errors, and imperfect ontology alignments frequently lead to unintended consequences. Among the most common consequences of flawed alignments is the violation of the *conservativity principle*.

Definition 1.1. Conservativity principle. The conservativity principle states that merging two ontologies starting from an alignment should not introduce any new subsumption or equivalence relations between entities originating from the same input ontology (JIMÉNEZ-RUIZ; GRAU; HORROCKS; BERLANGA, 2011).

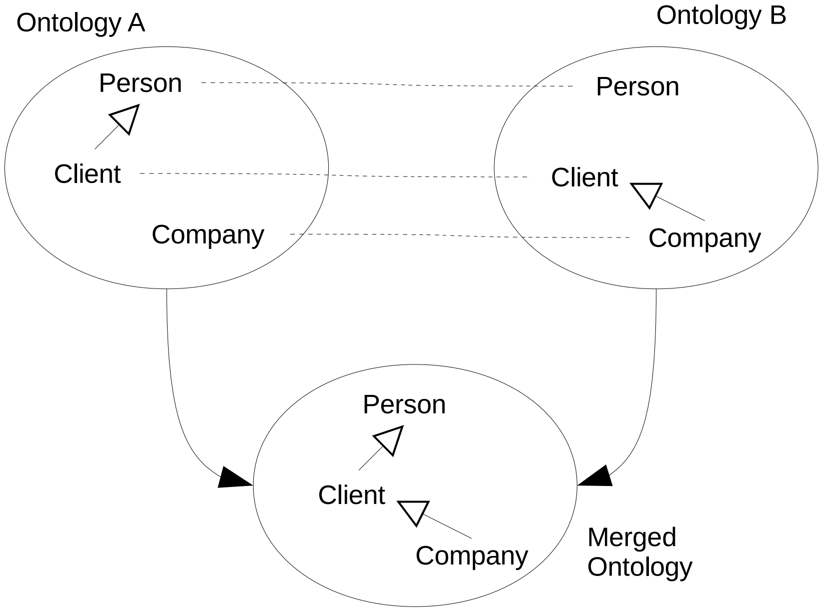
Definition 1.2. Subsumption. A subsumption relation holds between two entities *c* and *d* if and only if every instance of *d* is also an instance of *c*, in which case *c subsumes d*.

Definition 1.3. Equivalence. An equivalence relation holds between two entities if they share all their instances in every possible state of the world. That is, *c is equivalent to d* if and only if every instance of *c* is an instance of *d* and every instance of *d* is an instance of *c*. Therefore, equivalence implies and is implied by mutual subsumption.

Whenever the conservativity principle is violated, the resulting ontology does not preserve the intended meaning of the concepts specified in the original ontologies. Take as example an alignment that matches concepts *person*, *client* and *company* from ontologies *A* and *B*, where ontology *A* states that *person subsumes client* (that is, every client is a person) and ontology *B* states that *client subsumes company* (every company is a client), as shown in

Figure 1.1. If the ontologies are merged through the alignment, the resulting ontology will state that *person subsumes company*, that is, that every company is a person, which does not reflect the knowledge in either of the original ontologies. In fact, a query for *person* in the merged ontology will return every instance of *company* in the knowledge base, which does not agree with the return of the same query over any of the source ontologies separately. This disagreement shows that the conceptualization expressed in the resulting ontology is incompatible with those from the input ontologies.

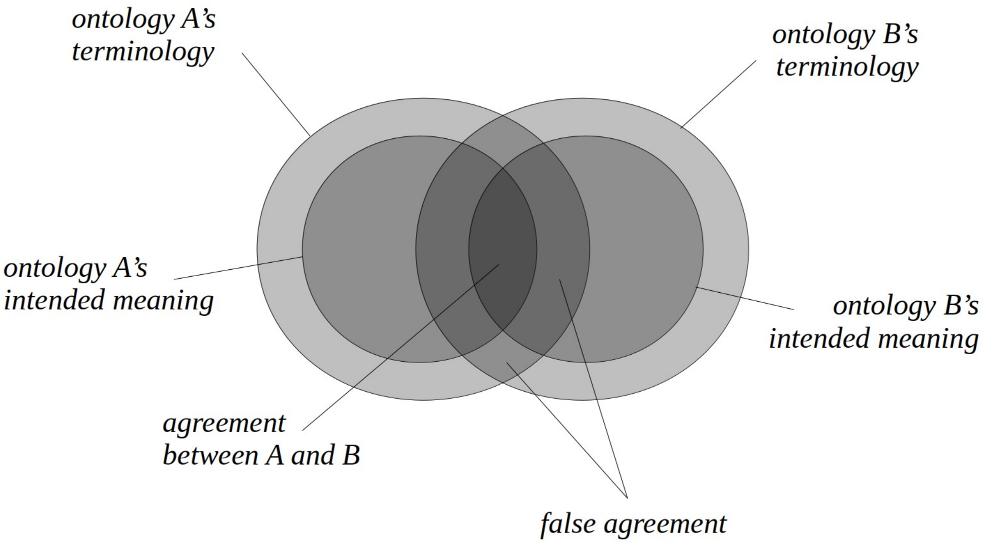
Figure 1.1– Example of mismatched concepts



Source: the author.

This incompatibility indicates that the alignment is matching concepts whose meaning is different in each ontology, i.e., it is equating concepts that are not the same. Such matches occur due to terminological or structural similarities between the concepts, which lead the domain expert or the matching algorithm to believe their meaning is the same even when it is not. Figure 1.2 depicts an abstraction of this problem. An alignment may cover any portion of the intersection between ontologies A and B. However, a large part of the intersection of the ontologies terminologies does not account for their conceptualizations, i.e., the intended meaning of the terms. The goal of this work is to improve ontology alignments by removing incorrect matches which originate from this false agreement. We take the violations of the conservativity principle as evidence of such incorrect matches.

Figure 1.2 – False agreement between ontologies A and B.



Source: the author.

In the previous example, the concept *client* in the first ontology refers specifically to personal clients, i.e., persons that are clients of some company, while in the second ontology the concept *client* refers to a broader range of entities, including corporate clients, that is, companies which are clients of other companies. Further, in the second ontology the concept *company* actually refers only to corporate clients, excluding suppliers, for example, while the concept *company* in the first ontology does not have this restriction. Therefore, the concept *client* does not have the same meaning in both ontologies, and the same holds for concept *company*. The introduction of new subsumption relations by the alignment, violating the conservativity principle, evidences this fact.

Another frequent kind of consequence from flawed alignments is the inconsistency of the merged ontology, that is, the presence of concepts that cannot be satisfied. While inconsistency reflects logical flaws in the alignment, conservativity violations reflect semantic flaws. As such, conservativity violations are harder for the user to perceive, and much of their danger lies in the fact that the resulting ontology upholds a divergent view of the world which is not immediately clear for the users. For the same reasons, non-conservative alignments are even more common than inconsistent alignments, i.e., alignments that result in inconsistent ontologies. The present dissertation deals only with the problem of detecting and correcting conservativity violations, while consistency issues fall outside the scope of this work.

It should be noted that there are distinct perspectives on conservativity violations, regarding if they indicate flaws in the alignment or in the source ontologies (Solimando, Jímenez-Ruiz, Guerrini, 2017). The latter perspective, adopted by Lambrix and Liu (2013) and Ivanova and Lambrix (2013), holds that the introduction of new subsumption relations in the merged ontology is an evidence of missing subsumption relations in the source ontologies. We assume the first perspective, i.e., that violations denote incorrectly matched concepts. This perspective reflects the fact that distinct ontologies are usually designed with different views on the world, an aspect that is ignored if the violations are taken as flaws in the ontology itself. The understanding of violations of conservativity as indicators of mismatches, on the other hand, emphasizes the differences in the conceptualizations accepted by each ontology by understanding that the conflicting sets of subsumption relations arise from such divergent outlooks on the world.

A final key aspect of conservativity principle violations is that they may be divided in two kinds: violations of subsumption conservativity and violations of equivalence conservativity. Violations of subsumption conservativity occur when new subsumption relations are introduced between concepts originating from the same source ontology. Violations of equivalence conservativity occur when new equivalence relations are introduced between concepts originating from the same source ontology.

Each kind of violation behaves slightly different. Violations of equivalence conservativity, for example, may be introduced either through the mapping of two or more entities in one of the input ontologies to a single entity in the other, or through the introduction of circular chains of subsumption relations. Due to this divergence in behavior, some approaches, such as in the work of Solimando, Jiménez-Ruiz and Guerrini (2014b, 2017), deal differently with each kind of violation. However, since equivalence implies and is implied by mutual subsumption, as previously mentioned, the approach proposed in the present dissertation does not explicitly differentiate violations of each kind.

The remaining of this dissertation is structured as follows. Chapter 2 presents basic notions related to ontologies and ontology engineering. Chapter 3 provides a brief introduction to category theory and its constructs. Chapter 4 discusses related work from the literature dealing with conservativity principle violations. Chapter 5 describes previous work on categories of ontologies and the formalization of ontology alignments. Chapter 6 presents our analysis of the conservativity of ontology alignments under category theory. Chapter 7 proposes an approach for correcting conservativity violations, whose experimental evaluation is presented and discussed in Chapter 8. Finally, Chapter 9 presents a brief conclusion.

2 ONTOLOGY

The word “ontology” frequently takes one of three distinct meanings. With a capital “O”, “Ontology” refers to the branch of philosophy that studies the fundamental nature of reality, dating back to Aristotle. Still in the realm of Philosophy, “ontology”, with a lowercase “o”, describes a specific system of categories that accounts for some worldview. In Computer Science, however, the prevailing meaning of “ontology” is that of a computational artifact that models the structure of a certain part of reality (GUARINO; OBERLE; STAAB, 2009).

Definition 2.1. Ontology. A formal, explicit specification of a shared conceptualization (STUDER; BENJAMINS; FENSEL, 1998).

Definition 2.1 defines ontology in the sense of a computational artifact. In this definition, “formal” means machine-readable, “explicit” reflects that all knowledge in the ontology should be defined explicitly and not depend on a previous understanding of any user, “shared” means that the information in the ontology should come from a understanding of the domain that is consensual among a community of people and not private to some individual, and “conceptualization” refers to an abstract model of a portion of reality. The idea of “conceptualization” is thus closely related to the philosophical meaning of “ontology”.

2.1 Universals, Classes and Particulars

Ontologies, as specifications of abstract models of reality, deal mainly with *universals*, that is, types or concepts that generalize the common properties of similar entities in the world. Knowledge bases built upon such ontologies contain additional information regarding the *particulars* or individuals which manifest such universals. A particular that manifests some universal is said to instantiate it. For example, a person called *John* is a particular that instantiates the universal *Human*. The following definitions are based on Guarino, Oberle and Staab (2009).

Definition 2.2. Entity. The term “entity” is the most general term to denote beings, including both universals and particulars.

Definition 2.3. Universal. An entity that may be instantiated, roughly corresponding to a predicate in first-order logic.

Definition 2.4. Particular. An entity that cannot possibly be instantiated.

Ontologies are intended to represent knowledge that is independent from any specific state of affairs in the world. Therefore, universals are ideally defined *intensionally*, i.e., by specifying their meaning. Every universal carries a principle of application that guides the judgement on whether a specific particular realizes that universal. The intensional definition allows that the same universal may be manifested on different particulars depending on the world state.

In any specific world state, it is possible to find all instances of a universal and group them in a set. This comprehensive listing of all members of the set is an *extensional* definition that is only valid for a single state of affairs, unlike the intensional definitions based on the meaning of universals. Such extensional groups are called *classes*. The relation of subsumption between universals, discussed in Chapter 1, raises a subclass relation between the corresponding classes of the related universals – that is, if *a* subsumes *b*, the extensional class of *a* is a superclass of the extensional class of *b.*

In the remainder of this work, we will use the terms “concepts” to refer to universals which correspond to unary predicates, and “relations” to refer to universals corresponding to binary predicates. In doing so, we abstract from representational issues and consider both the intensional and the extensional aspect of universals. Since ontologies are not ultimately concerned with particulars, we will often refer generically to the concepts and relations in an ontology as entities. We refrain from using the term “universal” since some ontology representation languages do not account for the intensional aspects inherent to universals, representing only the extensional classes. This is the case of the Web Ontology Language, as we shall discuss in Section 2.2.

Definition 2.5. Concept. A universal corresponding to a unary predicate in first order logic.

Definition 2.6. Relation. A universal corresponding to a binary predicate in first order logic.

2.2 The Web Ontology Language

The Web Ontology Language (OWL) is actually a family of languages for the representation of knowledge designed specifically for the development of ontologies (ANTONIOU; HARMELEN, 2009). It was further conceived to be adopted in the context of the Semantic Web, where it is currently among the most popular ontology representation languages.

Entities represented in OWL fall into three broad groups: individuals, classes and properties. Individuals are particular objects (e.g., John, a specific apple), classes are categories of individuals (e.g., person, apple, employee) and properties are relations between individuals (e.g., married to) or from individuals to data values (e.g., has age). Subclass relations hold between classes and subproperty relations hold between properties.

Definition 2.7. Class. The extensional aspect of a concept.

Definition 2.8. Property. The extensional aspect of a relation.

Thus, classes and properties represent the extensional groupings that arise from concepts and relations, respectively, in a determined world state. Therefore, whenever an entity is said to be an “ancestor” or “descendant” of another, that is, that a subsumption relation holds between such entities, it means that the appropriate subclass or subproperty relation holds between the underlying extensions of those entities.

The representation constructs in OWL have been selected to balance the semantic expressiveness required by typical applications with the computational efficiency for the supported reasoning tasks. Among such tasks, OWL supports the classification of individuals (i.e., finding to which class an individual belongs), determination of class equivalencies and consistency verification. For these reasons, when it comes to the practical implementation and evaluation of the algorithms proposed in this dissertation (Chapters 6 and 7), the ontology representation language in use is OWL.

3 CATEGORY THEORY

Category theory is a branch of mathematics that studies the structure present in systems of composable relations between mathematical objects. These relations are called “morphisms” and are abstractions of structure-preserving mappings from several distinct branches of mathematics, such as functions, graph homomorphisms and linear transformations. Morphisms are the basic primitives upon which category theory is grounded.

Definition 3.1. Category. A category is a quadruple $C = (O, hom, id, \circ)$, consisting of:

- a class O of objects,
- for each pair (A,B) of objects a set $hom(A,B)$ of morphisms from A to B,
- for each object A, an identity morphism $id_A{:}A \rightarrow A$, and
- a composition operator $\circ$ associating each pair of morphisms $f{:}A \rightarrow B$ and $g{:}B \rightarrow C$ to a composite morphism $g \circ f{:}A \rightarrow C$,

such that (1) the composition operator is associative, i.e., for any three morphisms $f{:}A \rightarrow B$, $g{:}B \rightarrow C$ and $h{:}C \rightarrow D$, $h \circ (g \circ f) = (h \circ g) \circ f$; (2) identities are neutral regarding composition, that is, for any morphism $f{:}A \rightarrow B$, $id_B \circ f = f = f \circ id_A$; and (3) the sets $hom(A,B)$ are pairwise disjoint (ADÁMEK; HERRLICH; STRECKER, 1990).

3.1 Limits and Colimits

A diagram in a category is a selection of some of its objects and morphisms. A source for a diagram is a pair (x,f_i) consisting of an object x and a family of morphisms $f_i{:}x \rightarrow d_i$ with domain x and codomain indexed by the diagram, i.e., a group of morphisms from x to each object in the diagram. If, for any morphism $g{:}d_i \rightarrow d_j$ in the diagram, the triangle formed by g, f_i and f_j commutes, that is, $g \circ fi = fj$, then the source (x,f_i) is called a cone. If (x,f_i) is a terminal cone, that is, if for every other cone (x',f_i') there exists a unique morphism $h{:}x' \rightarrow x$ such that the resulting diagram commutes, (x,f_i) is a limit.

Every category-theoretic construct has a dual built by reversing the direction of the morphisms, i.e., switching each morphism’s domain for its codomain. Thus, the dual to a source is a sink, a pair (x,f_i) consisting of an object x and a family of morphisms $f_i{:}d_i \rightarrow x$ with codomain x and domain indexed by the diagram, that is, a group of morphisms from each object in the diagram to x. A commutative sink is a cocone, which is dual to a cone, and the initial cone is a colimit, i.e., the dual to a limit.

Figure 3.1 – A limit and a colimit.

$$\begin{tikzcd}
d_i \arrow[rr, "g"] & & d_j \\
 & x \arrow[ul, "f_i"'] \arrow[ur, "f_j"] & \\
 & x' \arrow[uul, "f'_i"] \arrow[uur, "f'_j"'] \arrow[u, dotted, "h"'] &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & x' & \\
 & x \arrow[u, dotted, "h"] & \\
d_i \arrow[rr, "g"'] \arrow[ur, "f_i"'] \arrow[uur, "f'_i"] & & d_j \arrow[ul, "f_j"] \arrow[uul, "f'_j"']
\end{tikzcd}$$

Figure 3.1 depicts a cone and a limit (on the left) and a cocone and a colimit (on the right). Limits and colimits have meaningful interpretations in several categories. For example, a cartesian product is the limit of a diagram containing two (or more) objects (and no morphisms) in the category of sets.

3.2 Pullbacks and Pushouts

This dissertation is particularly interested in a specific type of limit and its dual, which are pullbacks and pushouts, respectively. Pullbacks are limits of diagrams containing two morphisms *f:A→C* and *g:B→C* with a shared codomain. Pushouts are colimits of diagrams containing two morphisms *f:A→B* and *g:A→C* with a shared domain. Figure 3.2 depicts a pullback on the left and a pushout on the right.

Figure 3.2 – A pullback and a pushout.

$$\begin{tikzcd}
 & C & \\
A \arrow[ur] & & B \arrow[ul] \\
 & x \arrow[ul] \arrow[ur] & \\
 & x' \arrow[uul] \arrow[uur] \arrow[u, dotted] &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & x' & \\
 & x \arrow[u, dotted] & \\
B \arrow[ur] \arrow[uur] & & C \arrow[ul] \arrow[uul] \\
 & A \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

In a category of ontologies with total mappings as morphisms, according to Zimmerman, Krötzsch, Euzenat and Hitzler (2006), pushouts designate the merging of two ontologies from an alignment between them. And according to Cafezeiro and Haeusler (2007), pullbacks denote similarity searches between two ontologies in the context of a broader one (that is, their intersection) in the same category. This category is further discussed in Chapter 5.

4 RELATED WORK

Jiménez-Ruiz, Grau, Horrock and Berlanga (2011) introduced the conservativity principle for ontology alignments based on the notion of conservative extension in description logics. They also proposed an algorithm for correcting equivalence conservativity violations by checking for direct mappings between two different concepts in one ontology to a single concept in the other. To the extent of our knowledge, this is the only existing approach for correcting conservativity violations that does not requires the merging of the aligned ontologies. However, it also does not account for subsumption conservativity violations, and ignores the cases where new equivalence relations arise from the introduction of circular chains of subsumption relations.

Even before the term “conservativity” was introduced, Meilicke (2006) studied the same property under the name of “stability” and proposed an algorithm that checks for the stability of mappings from individual ontologies to the merged ontology. The verification process iterates over the classes in the local ontology and checks if their superclasses are the same in the local context, i.e., in the separate ontologies, and in the distributed context, that is, in the merged ontology. The author does not, however, present an algorithm to correct the detected violations.

In another work prior to the proposal of the conservativity principle, Jean-Marie, Shinoroshita and Kabuka (2009) proposed the ASMOV algorithm, standing for Automated Semantic Matching of Ontologies with Verification. As the name suggests, ASMOV is an algorithm to compute an alignment between ontologies. As a step in the alignment construction, ASMOV performs a semantic verification process. This process merges the aligned ontologies and then selects iteratively two pairs of matched entities and verifies several kinds of inference in the merged ontology, including multiple-entity correspondences (i.e., when a single entity is mapped to several), crisscross correspondences (when equivalence relations are introduced due to the matching of pairs of entities with reverse subsumption relations in each ontology), disjointness-subsumption contradictions (introduction of subsumption relations between disjoint entities), subsumption and equivalence incompleteness (introduction of subsumption and equivalence relations in general) and domain and range incompleteness (mismatch of the domains of matched properties). All checks but the last refer to the conservativity principle.

All the approaches discussed thus far take the perspective that conservativity violations reflect flaws in the ontology alignment. Lambrix and Liu (2013), on the other hand, hold that the violations evidence flaws in the ontologies themselves (Chapter 1 contains a brief discussion on both perspectives). Similarly to previous works, they propose to compute the merge of a network of ontologies and check if every subsumption relation in the merged ontology is also present in the respective source ontology. However, the authors take violations to indicate missing subsumption relations in the source ontologies, and accordingly the correction of said violations consists of the insertion of new subsumption relations in the ontologies. The correction process includes the selection, for each missing subsumption relation "*a subsumes b*", of a concept *c* that subsumes *a* and does not subsume *b* and a concept *d* subsumed by *b* that is not subsumed by *a* and creating a new subsumption relation "*c subsumes d*". The user chooses the subsumption relation to be created from a list of all possible choices – thus, the process is not fully automated. Ivanova and Lambrix (2013) modified this approach by submitting the detected violations to a domain expert for validation regarding whether the violations indicate missing subsumption relations or mismatches in the alignment, taking into account both perspectives on the interpretation of conservativity violations.

The previous approaches all rely on comparing all subsumption relations in the individual ontologies to the ones in the merged ontology. Distinctly, Solimando, Jiménez-Ruiz and Guerrini (2014a) reduced the problem of detecting subsumption conservativity violations to the detection of unsatisfiable concepts. In order to achieve this reduction, they followi the assumption of disjointness, that is, that all concepts that do not share subsumees are disjoint. We believe, however, that the assumption of disjointness is not reasonable, since many ontologies allow the simultaneous instantiation of concepts that do not share subsumees, mainly due to the huge number of concepts that would need to be made explicit otherwise. Take as example a small ontology with concepts *man*, *woman*, *teenager*, *adult*, *elderly*, *student*, *artist*, *retail worker*, *farmer*. In order to make explicit every possible combination of non-disjoint concepts, even if we assume that the sets *{man*, *woman}*, *{teenager, adult, elderly}* and *{student, artist, retail worker, farmer}* are each pairwise disjoint, would require the inclusion of at least 61 new concepts, including concepts such as *teenager woman, elderly farmer* and *adult male artist*. This means that the small ontology would grow more than five times. If we do not assume that the concepts in each set are disjoint, this number grows even bigger. Another important aspect of the mentioned work is that locality-based modules are extracted from the aligned ontologies and codified as Horn propositional theories prior to its extension with additional disjointness axioms and the detection of unsatisfiable concepts.

Later, Solimando, Jiménez-Ruiz and Guerrini (2014b, 2017) extended their previous approach by using it in conjunction with an algorithm that detects equivalence conservativity violations trough the search of loops in directed graphs representing the ontologies, where nodes are concepts and edges are subsumption relations. The algorithm proposed was implemented as an extension of the LogMap ontology matching and mapping repair system (Jiménez-Ruiz, Grau, 2011). Due to the availability of empirical results and of a current implementation, as well as being the most recent proposal on the subject, we have chosen this work to serve as basis of comparison to the algorithm proposed in this dissertation. Chapter 8 presents the results in detail.

Finally, Antunes, Rademaker and Abel (2019) analyzed conservativity violations under category theory and proposed an algorithm for detecting such violations without necessity of merging the aligned ontologies. This analysis was the first step for the formalization discussed in the beginning of Chapter 6.

Table 4.1 – Comparison of alignment conservativity validation approaches

Approach	*Requires merge*	*Violation as flaw in*	*Automatic*	*Extract modules*
Meilicke (2006)[1]	Yes	Alignment	Yes	No
Jean-Marie, Shinoroshita and Kabuka (2009)	Yes	Alignment	Yes	No
Jiménez-Ruiz, Grau, Horrock and Berlanga (2011)[2]	No	Alignment	Yes	Yes
Lambrix and Liu (2013)	Yes	Ontologies	Semi	No
Ivanova and Lambrix (2013)[3]	Yes	Either	Semi	No
Solimando, Jiménez-Ruiz and Guerrini (2014a)[4] [5]	Yes	Alignment	Yes	Yes
Solimando, Jiménez-Ruiz and Guerrini (2014b, 2017)[4]	Yes	Alignment	Yes	Yes
Antunes, Rademaker and Abel (2019)[1]	No	Alignment	Yes	No

Source: the author.

Table 4.1 presents a summary of the approaches discussed in this chapter. As we have seen, most approaches require the aligned ontologies to be merged as part of the validation of conservativity. Additionally, violations are widely taken as flaws in the alignment, although some authors acknowledge them as flaws in the aligned ontologies instead. We follow the trend of regarding violations as indications of flaw in the alignments and take a “better safe

---

[1] Does not compute repair

[2] Only considers violations of equivalence conservativity

[3] The user defines whether each violation indicates a flaw in the alignment or in the ontologies

[4] Follows the assumption of disjointness

[5] Only considers violations of subsumption conservativity

than sorry” approach and aim to remove every violation. Differently from previous works, our method does not, however, relies on merging the input ontologies, leading to an improvement in efficiency.

5 ONTOLOGY ALIGNMENTS IN CATEGORIES OF ONTOLOGIES

Traditionally, authors define categories of ontologies with total mappings as morphisms. Alignments between ontologies seldom match the entirety of any one of the aligned ontologies, mapping only subsets of the concepts and relations in each ontology. Therefore, it is necessary to formalize such alignments as structures that are more complex than individual morphisms.

Bench-Capon and Malcom (1999) define relations between two ontologies O_1 and O_2 as a structure composed of a pair of morphisms $x_1 : O \rightarrow O_1$ and $x_2 : O \rightarrow O_2$ with a shared domain O. Thus, entities in the aligned ontologies are matched by being mapped from the same entity in O.

Figure 5.1 – Composition, intersection and union of V-alignments

$$\begin{tikzcd}
O_1 & & O_2 & & O_3 \\
& A \arrow[ul, "\alpha_1"] \arrow[ur, "\alpha_2"'] & & B \arrow[ul, "\beta_2"] \arrow[ur, "\beta_3"'] & \\
& & C \arrow[ul, "f_A"] \arrow[ur, "f_B"'] \arrow[uull, dashed, bend left, "\gamma_1"] \arrow[uurr, dashed, bend right, "\gamma_3"'] & &
\end{tikzcd}$$

$$\begin{tikzcd}
O_1 & & O_2 \\
A \arrow[u, "f_1"] \arrow[urr, bend left, "f_2"] & & B \arrow[u, "g_2"'] \arrow[ull, bend right, "g_1"'] \\
& C \arrow[ul, "k_A"] \arrow[ur, "k_B"'] \arrow[uul, dashed, bend left, "h_1"] \arrow[uur, dashed, bend right, "h_2"'] &
\end{tikzcd}
\qquad
\begin{tikzcd}
O_1 & & O_2 \\
& C' \arrow[ul, dashed, "u_1"'] \arrow[ur, dashed, "u_2"] & \\
A \arrow[uu, "f_1"] \arrow[uurr, bend left, "f_2"] \arrow[ur, "i_A"] & & B \arrow[uu, "g_2"'] \arrow[uull, bend right, "g_1"'] \arrow[ul, "i_B"'] \\
& C \arrow[ul, "k_A"] \arrow[ur, "k_B"'] &
\end{tikzcd}$$

Source: Zimmerman, Krötzsch, Euzenat and Hitzler (2006).

Later, Zimmerman, Krötzsch, Euzenat and Hitzler (2006) named such structures as V-alignments, due to their shapes. The authors defined the merging of aligned ontologies as a pushout over the V-alignment, and also defined three operations over V-alignments: composition, intersection and union. Figure 5.1 depicts those operations.

Definition 5.1. V-alignment. A V-alignment between ontologies O_1 and O_2 is a triple $(A, \alpha_1 : A \rightarrow O_1, \alpha_2 : A \rightarrow O_2)$, where A is an ontology and α_1, α_2 are total mappings from A into the aligned ontologies.

Definition 5.2. V-alignment composition. The composition of two V-alignments $(A, \alpha_1{:}A \rightarrow O_1, \alpha_2{:}A \rightarrow O_2)$ and $(B, \beta_2{:}B \rightarrow O_2, \beta_3{:}B \rightarrow O_3)$ is the limit of the diagram containing both V-alignments.

Definition 5.3. V-alignment intersection. The intersection of two V-alignments $(A, f_1{:}A \rightarrow O_1, f_2{:}A \rightarrow O_2)$ and $(B, g_1{:}B \rightarrow O_1, g_2{:}B \rightarrow O_2)$ between two ontologies O_1 and O_2 is the limit of the diagram containing both V-alignments.

Definition 5.4. V-alignment union. The union of two V-alignments $(A, f_1{:}A \rightarrow O_1, f_2{:}A \rightarrow O_2)$ and $(B, g_1{:}B \rightarrow O_1, g_2{:}B \rightarrow O_2)$ is the pushout over the diagram containing A, B, their intersection C and the injections $k_A{:}C \rightarrow A$ and $k_B{:}C \rightarrow B$.

V-alignments are sufficient to represent alignments where entities are matched by equivalence relations. However, they are not suitable for cases where the aligned ontologies are related indirectly and entities must be matched by subsumption relations instead. Zimmerman, Krötzsch, Euzenat and Hitzler (2006) answer that issue with W-alignments, that relate the aligned ontologies through bridge ontologies that allow the inclusion of new information not present in the source ontologies. An example would be the assertion that the concept *person* in ontology O_1 subsumes the concept *woman* in ontology O_2. This assertion would be represented as a subsumption relation *c subsumes d* between concepts *c* and *d* in the bridge ontology such that *c* is mapped to *person* in O_1 and *d* is mapped to *woman* in O_2.

Definition 5.5. W-alignment. A W-alignment between ontologies O_1 and O_2 is a triple (B, A_1, A_2), where B is a bridge ontology and A_1, A_2 are V-alignments between the bridge ontology and O_1 and O_2, respectively.

The merge of ontologies through a W-alignment is a colimit under the W-alignment – which is equivalent to the pushout over both the separate pushouts over each leg of the "W", as depicted in Figure 5.2.

Figure 5.2 – Merging with W-alignments

$$\begin{tikzcd}
& & M & & \\
& O_1^+ \arrow[ur, dashed] & & O_2^+ \arrow[ul, dashed] & \\
O_1 \arrow[ur, dashed] & & B \arrow[ul, dashed] \arrow[ur, dashed] & & O_2 \arrow[ul, dashed] \\
& A_1 \arrow[ul] \arrow[ur] & & A_2 \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Source: Zimmerman, Krötzsch, Euzenat and Hitzler (2006).

Section 6.1 provides a formalization of conservativity in V-alignments, along with an analysis of conservativity under V-alignments operations. Section 6.2 presents a formalization of conservativity under W-alignments.

Cafezeiro and Haeusler (2007) introduced further constructions in a category of ontologies. The authors formalized the similarity of two ontologies in the context of a broader ontology as a pullback from two ontology mappings, and the “copy” of an ontology with hidden sensitive information as an equalizer, that is, a limit over a diagram containing two morphisms with same domain and codomain.

6 DETECTING VIOLATIONS OF CONSERVATIVITY WITH CATEGORY THEORY

For the scope of this work, we define a category of ontologies *Ont*.

Definition 6.1. *Ont*. A category $Ont = (O_{Ont}, hom_{Ont}, id_{Ont}, \circ_{Ont})$, consisting of:

- a class of objects O_{Ont}, where each object is an ontology in the form of a pair (E,S), such that
  - E is a set of entities in the ontology, that is, the disjoint union of
    - C, a set of concepts, and
    - R, a set of relations;
  - $S \subseteq E \times E$ is a transitive, reflexive and antisymmetric subsumption relation, subject to
    - concepts can’t subsume relations and vice versa, that is, $\forall\, c \in C, r \in R, (c, r) \notin S \wedge (r, c) \notin S$;
- for each pair (A,B) of objects, a set hom_{Ont} of functions $f(x){:}E_A \rightarrow E_B$ as morphisms, such that
  1. concepts are mapped to concepts, i.e., $\forall\, e \in E_A,\ e \in C_A \leftrightarrow f(e) \in C_B,$
  2. relations are mapped to relations, i.e., $\forall\, r \in E_A,\ r \in R_A \leftrightarrow f(r) \in R_B,$
  3. subsumption is preserved, i.e., $\forall\, e, d \in E_A,\ (e, d) \in S_A \rightarrow (f(e),f(d)) \in S_B,$
  4. relations are preserved, i.e., $\forall\, r \in R_A, \forall\, c, d \in C_A,\ (c, d) \in r \rightarrow (f(e),f(d)) \in f(r)$;
- for each object A, an identity function over the entity set as identity morphism id_A, i.e., $\forall\, e \in E_A, id_A(e) = e$;
- the composition operator $\circ_{Ont}$ is usual function composition.

It is important to note that the ontology is not the same as its representation in OWL 2 or other ontology representation language. Usually, many subsumption relations are not explicit in the representation and must be inferred from the ontology’s axioms by some OWL 2 reasoner. However, every entity and subsumption relation in the ontology must be in the E and S sets, respectively. Therefore, whenever this dissertation refers to an ontology, it is describing the sets of entities and subsumption relations that are explicit in the representation along with all the additional entities and subsumption relations that can be inferred from reasoning tasks, and not those that are explicit in the representation.

Proposition 6.1. *Ont* is a category.

Proof. In order for *Ont* to be a category, (a) there must exist, for each pair of morphisms $f{:}A \rightarrow B$ and $g{:}B \rightarrow C$ a composite morphism $g \circ f{:}A \rightarrow C$, (b) composition must be associative, i.e., for any three morphisms $f{:}A \rightarrow B$, $g{:}B \rightarrow C$ and $h{:}C \rightarrow D$, $h \circ (g \circ f) = (h \circ g) \circ f$, and (c) for every object there must exist an identity morphism that is neutral with respect to composition, that is, for any morphism $f{:}A \rightarrow B$, $id_B \circ f = f = f \circ id_A$. Composition in *Ont* is usual function composition, which always exist and is associative. However, the composite must additionally follow rules (1) to (3). Take two morphisms $f{:}A \rightarrow B$ and $g{:}B \rightarrow C$. Since f follows (1), every concept in C_A is mapped to a concept in C_B and every relation in R_A is mapped to a relation in R_B. Since g also follows (1), every concept in C_B is mapped to a concept in C_C and every relation in R_B is mapped to a relation in R_C. Therefore, $g \circ f$ maps every concept in C_A to a concept in C_C and every relation in R_A to a relation in R_C, respecting (1). Since f preserve relations, according to (3), for any relation r in R_A that holds between concepts c and d in C_A, $f(r)$ must hold between $f(c)$ and $f(d)$. Since g also preserves relations, we have that $g(f(r))$ holds between $g(f(c))$ and $g(f(d))$, i.e., $g \circ f$ also follows (3). Similarly, since both f and g preserve subsumption, as stated by (2), for any pair of entities e and d in C_A, if e subsumes d, then $f(e)$ subsumes $f(d)$ and $g(f(e))$ subsumes $g(f(d))$, observing rule (2). Finally, identities simply map each concept and relation to itself. As required for identities, such mappings are neutral on composition, since for any $f{:}A \rightarrow B$, $g{:}B \rightarrow C$ and $e \in E_A$, $i_A(f(e)) = f(e)$ and $g(i_A(e)) = g(e)$.

6.1 Conservativity in V-Alignments

Given two ontologies A and B, a V-alignment $(V, f_A{:}V \rightarrow A, f_B{:}V \rightarrow B)$ is non-conservative if merging both ontologies via the alignment (a pushout, as discussed in Section 4.1) causes new subsumption relations to be introduced between two entities originated from the same source ontology. Since f_A and f_B are morphisms in *Ont* and, consequently, preserve subsumption, V cannot have a subsumption relation that is not in both A and B. Any new subsumption relation between entities from the same source ontology must therefore hold in the other aligned ontology.

Take the sub-ontologies A^V and B^V that include all aligned entities (and no other entity) and every subsumption relation between those entities in A and B respectively. If the alignment is conservative, A^V and B^V share all subsumption relations and it is possible to map both sub-ontologies onto each other preserving subsumption and matching the aligned entities from both ontologies. These mappings are morphisms in *Ont*. If at least one of these morphisms do not exist, then the alignment is non-conservative (ANTUNES; RADEMAKER; ABEL, 2019).

In category-theoretical terms, the sub-ontology A^V (B^V), along with morphisms $g_A^V{:}A^V \rightarrow A$ and $h_A^V{:}A^V \rightarrow V^*$ ($g_B^V{:}B^V \rightarrow B$ and $h_B^V{:}B^V \rightarrow V^*$), is the pullback from the diagram with morphisms $j_A{:}A \rightarrow A^*$ ($j_B{:}B \rightarrow B^*$) and $k_A^V{:}V^* \rightarrow A^*$ ($k_B^V{:}V^* \rightarrow B^*$), where V^* and A^* (B^*) are the extensions of V and A (B), respectively, with additional subsumption relations such that every entity subsumes each other. The morphism j_A (j_B) is the trivial inclusion that maps each entity in A (B) to its match in A^* (B^*) and k_A^V (k_B^V) is the morphism that makes the diagram with f_A^V, j_A (f_B^V, j_B) and the trivial inclusion $j_V{:}V \rightarrow V^*$ commute. Figure 6.1 shows the full diagram with all relevant objects and morphisms.

Since A^V (B^V) is the pullback of the diagram described and V along with morphisms f_A^V (f_B^V) and j_V is a cone for the same diagram, there exists a single morphism $f_A^V{:}V \rightarrow A^V$ ($f_B^V{:}V \rightarrow B^V$) such that the diagram commutes, that is, the mapping of entities from V into the sub-ontologies is consistent with the mapping into the original ontologies. If the alignment is conservative, there are two morphisms $m_A^V{:}A^V \rightarrow B^V$ and $m_B^V{:}B^V \rightarrow A^V$ such that $m_A^V \circ f_A^V = f_B^V$ and $m_B^V \circ f_A^V = f_B^V$. In the remainder of this section, the following equations will be useful:

(1) $m_A^V \circ f_A^V = f_B^V$

(2) $h_A^V \circ f_A^V = j_V = h_B^V \circ f_B^V$

(3) $h_A^V \circ f_A^V = h_B^V \circ m_A^V \circ f_A^V$ *from (1) and (2)*

(4) $h_A^V = h_B^V \circ m_A^V$ *removing f_A^V from (3)*

Figure 6.1 – Diagram for conservativity in V-alignments.

$$\begin{tikzcd}
 & A^* & & B^* & \\
A \arrow[ur, "j_A"] & & V^* \arrow[ul, "k_A^V"'] \arrow[ur, "k_B^V"] & & B \arrow[ul, "j_B"'] \\
 & A_V \arrow[ul, "g_A^V"] \arrow[ur, "h_A^V"] \arrow[rr, bend left, "m_A^V"] & & B_V \arrow[ul, "h_B^V"'] \arrow[ur, "g_B^V"'] \arrow[ll, bend left, "m_B^V"] & \\
 & & V \arrow[ul, "f_A^V"] \arrow[ur, "f_B^V"'] & &
\end{tikzcd}$$

It is noteworthy that this approach is enough to cover violations of equivalence conservativity as well as violations of subsumption conservativity. Since subsumption is reflexive, if two entities e and d in V are mapped to a single entity in A (i.e., $f_A(e) = f_A(d)$) and to different entities in B, the subsumption relations $(f_A^V(e), f_A^V(d))$ and $(f_A^V(d), f_A^V(e))$ will be present in A', while B' will not contain both matching relations unless $f_B(e) = f_B(d)$.

The following subsections present analyses of the behavior of conservativity under the V-alignment operations defined by Zimmerman, Krötzsch, Euzenat and Hitzler (2006): intersection, union and composition. Subsection 6.1.4 summarizes the findings.

6.1.1 Conservativity under V-Alignment Intersection

Given two V-alignments $(V, f_A^V{:}V \to A, f_B^V{:}V \to B)$ and $(U, f_A^U{:}U \to A, f_B^U{:}U \to B)$, their intersection $(V \cap U, f_A^{V\cap U}{:}V \cap U \to A, f_B^{V\cap U}{:}V \cap U \to B)$ is the limit for the diagram containing both alignments. If both V and U are non-conservative, there is no way of knowing a priori if $V \cap U$ is conservative or not. Take as example the case of an empty intersection or of an intersection with a single entity, which are trivially conservative since no new subsumption relation can be introduced between a single entity – subsumption is already reflexive. Another trivial case is when $V = U$. In this case, the intersection $V \cap U$ is also equal to V, and therefore is non-conservative if V is non-conservative. That is, when both alignments are non-conservative, the conservativity of the intersection is not certain and must be studied on a case by case basis, being necessary to compute the intersection and to analyze it independently. On the other hand, if either V or U are conservative, the intersection is always conservative.

Proposition 6.2. The intersection of a conservative V-alignment with another V-alignment (conservative or otherwise) is a conservative V-alignment.

Proof. Take the intersection $(V \cap U, f_A^{V\cap U}{:}V \cap U \to A, f_B^{V\cap U}{:}V \cap U \to B)$ and suppose that V is conservative. The extension of $V \cap U$ that is totally connected by subsumption, $V \cap U^*$, is the limit for the diagram containing $k_A^V{:}V^* \to A^*$, $k_B^V{:}V^* \to B^*$, $k_A^U{:}U^* \to A^*$ and $k_B^U{:}U^* \to B^*$. The sub-ontology $A^{V\cap U}$ is the pullback for the diagram with $j_A{:}A \to A^*$ and $k_A^{V\cap U}{:}V \cap U^* \to A^*$. Considering that $k_A^{V\cap U}$ factors trough k_A^V and $k_V^{V\cap U}{:}V \cap U^* \to V^*$, $(A^{V\cap U}, g_A^{V\cap U}{:}A^{V\cap U} \to A, k_V^{V\cap U} \circ h_A^{V\cap U}{:}A^{V\cap U} \to V^*)$ is a cone for the diagram containing $j_A{:}A \to A^*$ and k_A^V. Given that the sub-ontology A^V along with morphisms $g_A^V{:}A^V \to A$ and $h_A^V{:}A^V \to V$ is the pullback for this diagram, there exists a morphism $n_A{:}A^{V\cap U} \to A^V$ such that Equation (5) holds.

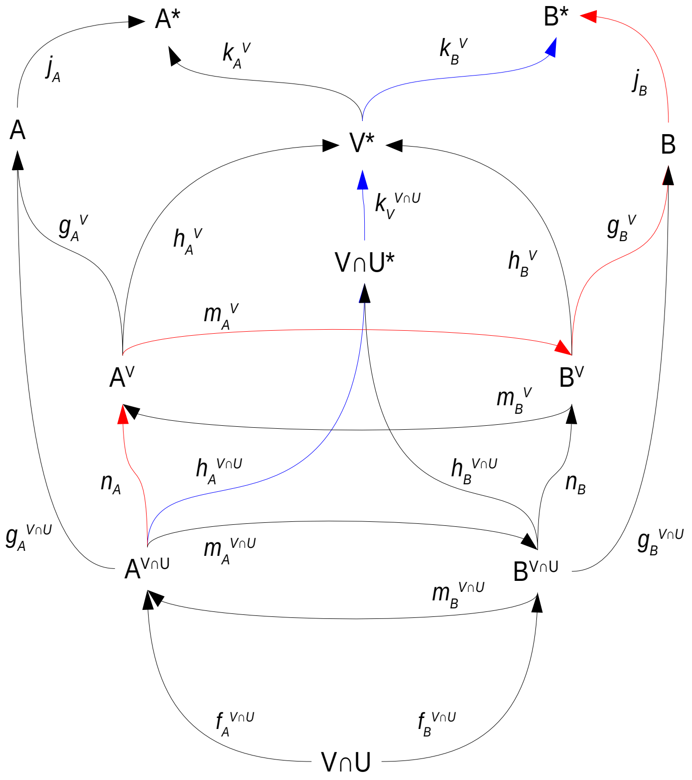
(5) $h_A^V \circ n_A = k_V^{V\cap U} \circ h_A^{V\cap U}$

Since V is conservative, there exists $m_A^V{:}A^V \to B^V$ such that Equations (1) through (4) hold. Therefore, $A^{V\cap U}$ along with $g_B^V \circ m_A^V \circ n_A{:}A^{V\cap U} \to B$ and $h_A^{V\cap U}{:}A^{V\cap U} \to V \cap U^*$ is a cone for the diagram with $j_B{:}B \to B^*$ and $k_B^{V\cap U}{:}V\cap U^* \to B^*$. Equations (6) through (9) delineate the steps to prove commutativity of the cone.

(6) $h_B^V \circ m_A^V \circ n_A = k_V^{V\cap U} \circ h_A^{V\cap U}$ *from (4) and (5)*
(7) $k_B^V \circ h_B^V \circ m_A^V \circ n_A = k_B^V \circ k_V^{V\cap U} \circ h_A^{V\cap U}$ *including k_B^V in (6)*
(8) $j_B \circ g_B^V \circ m_A^V \circ n_A = k_B^V \circ k_V^{V\cap U} \circ h_A^{V\cap U}$ *pullback commut. (7)*
(9) $j_B \circ g_B^V \circ m_A^V \circ n_A = k_B^{V\cap U} \circ h_A^{V\cap U}$ *simplification (8)*

Since $B^{V\cap U}$ is the pullback for the same diagram, there is a morphism $m_A^{V\cap U}{:}A^{V\cap U} \to B^{V\cap U}$. The existence of $m_B^{V\cap U}{:}B^{V\cap U} \to A^{V\cap U}$ can be proven in the same manner, exchanging A for B and vice versa. Thus, $V\cap U$ is conservative. Figure 6.2 depicts the relevant objects and morphisms, highlighting the morphisms involved in Equation (8), left side in red and right side in blue.

Figure 6.2 – Diagram for conservativity in V-alignment intersection.



6.1.2 Conservativity under V-Alignment Union

Given two V-alignments $(V,\ f_A^V{:}V \to A,\ f_B^V{:}V \to B)$ and $(U,\ f_A^U{:}U \to A,\ f_B^U{:}U \to B)$, their union $(V\cup U,\ f_A^{V\cup U}{:}V\cup U \to A,\ f_B^{V\cup U}{:}V\cup U \to B)$ is the pushout for the diagram containing

$f_V: V\cap U \to V$ and $f_U: V\cap U \to U$. Contrasting to the behavior of conservativity under intersection, it is impossible to know a priori if the union of two conservative alignments will be conservative or not. Take the example of the union of an alignment with itself, where $V=U$, which results in the original alignment. In this case, if V is conservative, the union is also conservative. However, if V and U are both single-entity alignments (thus, both conservative), such that V matches entity c in ontology A to entity d in ontology B and U matches the same entity c in A to a distinct and unrelated entity e in B. In this case, the union will match entity c to both d and e, introducing a new equivalence relation and being therefore non-conservative. If at least one of V and U is non-conservative, the union will also be non-conservative.

Proposition 6.3. The union of a non-conservative V-alignment with another V-alignment (conservative or otherwise) is a non-conservative V-alignment.

Proof. Take the union $(V\cup U,\ f_A^{V\cup U}: V\cup U \to A,\ f_B^{V\cup U}: V\cup U \to B)$ and suppose that V is non-conservative. The extension of $V\cup U$ that is totally connected by subsumption, $V\cup U^*$, along with morphisms $k_{V\cup U}^{V}: V^* \to V\cup U^*$ and $k_{V\cup U}^{U}: U^* \to V\cup U^*$ is the colimit for the diagram containing $k_V^{V\cap U}: V\cap U^* \to V^*$ and $k_U^{V\cap U}: V\cap U^* \to U^*$. Consequently, the sub-ontology A^V along with $k_{V\cup U}^{V} \circ h_A^V: A^V \to V\cup U^*$ and $g_A^V: A^V \to A$ is a cone for the diagram with $k_A^{V\cup U}: V\cup U^* \to A^*$ and $j_A: A \to A^*$. Since $(A^{V\cup U},\ g_A^{V\cup U}: A^{V\cup U} \to A,\ h_A^{V\cup U}: A^{V\cup U} \to V\cup U^*)$ is the pullback for the same diagram, there exists $n_A: A^V \to A^{V\cup U}$ such that Equation (10) holds.

(10) $h_A^{V\cup U} \circ n_A = k_{V\cup U}^{V} \circ h_A^V$

Suppose that the union $(V\cup U,\ f_A^{V\cup U}: V\cup U \to A,\ f_B^{V\cup U}: V\cup U \to B)$ is conservative. Then, there exists some $m_A^{V\cup U}: A^{V\cup U} \to B^{V\cup U}$ such that the following equations hold.

(11) $m_A^{V\cup U} \circ f_A^{V\cup U} = f_B^{V\cup U}$

(12) $h_A^{V\cup U} \circ f_A^{V\cup U} = j_{V\cup U} = h_B^{V\cup U} \circ f_B^{V\cup U}$

(13) $h_A^{V\cup U} \circ f_A^{V\cup U} = h_B^{V\cup U} \circ m_A^{V\cup U} \circ f_A^{V\cup U}$ *from (11) and (12)*

(14) $h_A^{V\cup U} = h_B^{V\cup U} \circ m_A^{V\cup U}$ *remov.* $f_A^{V\cup U}$ *from (13)*

Thus, A^V along with $g_B^{V\cup U} \circ m_A^{V\cup U} \circ n_A: A^V \to B$ and $h_A^V: A^V \to V^*$ is a cone for the diagram with $j_B: B \to B^*$ and $k_B^V: V \to B^*$. Equations (15) through (18) prove the commutativity of the cone.

(15) $h_B^{V\cup U} \circ m_A^{V\cup U} \circ n_A = k_{V\cup U}^{V} \circ h_A^V$ *from (10) and (14)*

(16) $k_B^{V\cup U} \circ h_B^{V\cup U} \circ m_A^{V\cup U} \circ n_A = k_B^{V\cup U} \circ k_{V\cup U}^{V} \circ h_A^V$ *include* $k_B^{V\cup U}$ *in (15)*

(17) $j_B \circ g_B^{V\cup U} \circ m_A^{V\cup U} \circ n_A = k_B^{V\cup U} \circ k_{V\cup U}^{V} \circ h_A^V$ *pullback commut. (16)*

(18) $j_B \circ g_B^{V\cup U} \circ m_A^{V\cup U} \circ n_A = k_B^{V} \circ h_A^V$ *simplification (17)*

Since B^V is the pullback for the same diagram, there must exist a morphism $m_A^V: A^V \to B^V$. The existence of $m_B^V: B^V \to A^V$ can be proven in the same manner, exchanging A

for B and vice versa. The existence of these two morphisms contradicts the assumption that V is non-conservative. Thus, it follows necessarily that, if V is non-conservative, the union is also non-conservative. The proof for U follows the same steps, switching V and U in the previous proof. Figure 6.3 shows the relevant objects and morphisms, highlighting the morphisms involved in Equation (17), left side in red and right side in blue.

Figure 6.3 – Diagram for conservativity in V-alignment union.

$$\begin{tikzcd}
& A^* & & & & B^* & \\
A \arrow[ur, bend left, "j_A"] & & & V\cup U^* \arrow[ull, "k_A^{V\cup U}"'] \arrow[urr, blue, "k_B^{V\cup U}"] & & & B \arrow[ul, red, bend right, "j_B"'] \\
& & & V^* \arrow[u, blue, "k_{V\cup U}^{V}"'] & & & \\
& A^{V\cap U} \arrow[uul, bend left, "g_A^{V\cup U}"] \arrow[uurr, bend left, "h_A^{V\cup U}"] \arrow[rrrr, red, bend left, "m_A^{V\cup U}"] & & & & B^{V\cup U} \arrow[uull, bend right, "h_B^{V\cup U}"'] \arrow[uur, red, bend right, "g_B^{V\cup U}"'] \arrow[llll, bend left, "m_B^{V\cup U}"] & \\
& A^V \arrow[u, red, "n_A"] \arrow[uurr, blue, bend right, "h_A^V"'] \arrow[uuul, bend left, "g_A^V"] \arrow[rrrr, bend left, "m_A^V"] & & & & B^V \arrow[u, "n_B"'] \arrow[uull, bend left, "h_B^V"] \arrow[uuur, bend right, "g_B^V"'] \arrow[llll, bend left, "m_B^V"] & \\
& & & V \arrow[ull, bend left, "f_A^V"] \arrow[urr, bend right, "f_B^V"'] & & &
\end{tikzcd}$$

By swaping $(V\cup U, f_A^{V\cup U}, f_B^{V\cup U})$ for any conservative alignment (X, f_A^X, f_B^X) and (V, f_A^V, f_B^V) for a cone over (X, f_A^X, f_B^X), it is easy to generalize this proof to demonstrate that any cone over a conservative alignment is also conservative. That is, any subset of the mappings in a conservative alignment is also conservative. Since both V and U are subsets of the mappings in the union, it follows that both must be conservative in order for $V\cup U$ to be conservative – although this is not a sufficient condition.

Corollary. Any subalignment of a conservative alignment is also conservative.

6.1.3 Conservativity under V-Alignment Composition

Given two V-alignments $(V, f_A^V{:}V\rightarrow A, f_B^V{:}V\rightarrow B)$ and $(U, f_B^U{:}U\rightarrow B, f_C^U{:}U\rightarrow C)$, their composite $(V\circ U, f_A^{V\circ U}{:}V\circ U\rightarrow A, f_C^{V\circ U}{:}V\circ U\rightarrow C)$ is the limit for the diagram containing both alignments. If any V or U is non-conservative, in order to verify the conservativity of the composite it is necessary to compute the composition and to check the resulting alignment. For example, if the sets of entities mapped by the alignments in B do not intersect with each other, the composite will be empty and therefore conservative. If, however, V is conservative and maps entities p and q in ontology A to entities r and s in B, and U is non-conservative and maps entities r and s in B to a single entity t in C, the resulting alignment will be non-conservative. There are also cases where the composite of two non-conservative alignments results in a conservative alignment. Take the previous example but assume that p and q are equivalent in A while r and s are not equivalent in B – in this case, both V and U are non-conservative, but the composite $V\circ U$, which maps both p and q to t, is conservative. However, if both V and U are conservative, the composite is necessarily conservative.

Proposition 6.4. The composite of two conservative V-alignments is a conservative V-alignment.

Proof. Take the composite $(V\circ U,\ f_A^{V\circ U}:V\circ U\rightarrow A,\ f_B^{V\circ U}:V\circ U\rightarrow C)$ and suppose that V and U are both conservative. The extension of $V\circ U$ that is totally connected by subsumption, $V\circ U^*$, along with morphisms $k_V^{V\circ U}:V\circ U^*\rightarrow V^*$ and $k_U^{V\circ U}:V\circ U^*\rightarrow U^*$ is the pullback from the diagram with $k_A^V:V^*\rightarrow A^*$, $k_B^V:V^*\rightarrow B^*$, $k_B^U:U^*\rightarrow B^*$ and $k_C^U:U^*\rightarrow C^*$. Thus, the sub-ontology $A^{V\circ U}$, along with morphisms $g_A^{V\circ U}:A^{V\circ U}\rightarrow A$ and $k_V^{V\circ U}\circ h_A^{V\circ U}:A^{V\circ U}\rightarrow V^*$, is a cone for k_A^V and $j_A:A\rightarrow A^*$. Since $(A^V,\ g_A^V:A^V\rightarrow A,\ h_A^V:A^V\rightarrow V)$ is the pullback for this diagram, there exists a morphism $n_A:A^{V\circ U}\rightarrow A^V$ such that Equation (19) holds.

(19) $h_A^V\circ n_A = k_V^{V\circ U}\circ h_A^{V\circ U}$

Since V is conservative, there exists $m_A^V:A^V\rightarrow B^V$ such that Equations (1) through (4) hold. Therefore, $A^{V\circ U}$ along with $g_B^V\circ m_A^V\circ n_A:A^{V\circ U}\rightarrow B$ and $h_A^{V\circ U}:A^{V\circ U}\rightarrow V\circ U^*$ is a cone for the diagram with $j_B:B\rightarrow B^*$ and $k_B^{V\circ U}:V\circ U^*\rightarrow B^*$. Equations (20) through (23) prove the commutativity of the cone.

(20) $h_B^V\circ m_A^V\circ n_A = k_V^{V\circ U}\circ h_A^{V\circ U}$ *from (4) and (19)*

(21) $k_B^V\circ h_B^V\circ m_A^V\circ n_A = k_B^V\circ k_V^{V\circ U}\circ h_A^{V\circ U}$ *inclusion of* k_B^V *(20)*

(22) $j_B\circ g_B^V\circ m_A^V\circ n_A = k_B^V\circ k_V^{V\circ U}\circ h_A^{V\circ U}$ *pullback commut. (21)*

(23) $j_B\circ g_B^V\circ m_A^V\circ n_A = k_B^{V\circ U}\circ h_A^{V\circ U}$ *simplification (22)*

Since $B^{V\circ U}$ is the pullback for the same diagram, there must exist a morphism $p_A^{V\circ U}:A^{V\circ U}\rightarrow B^{V\circ U}$ that makes the diagram commute. The existence of $p_B^{V\circ U}:B^{V\circ U}\rightarrow A^{V\circ U}$ can be proven in the same manner, exchanging A for B and vice versa. The proof of existence of $q_C^{V\circ U}:C^{V\circ U}\rightarrow B^{V\circ U}$ and $q_B^{V\circ U}:B^{V\circ U}\rightarrow C^{V\circ U}$ follows similar steps, exchanging V for U and A for C. Thus, the following equations hold.

(24) $f_A^{V\circ U} = p_B^{V\circ U} \circ f_B^{V\circ U}$

(25) $f_C^{V\circ U} = q_B^{V\circ U} \circ f_B^{V\circ U}$

(26) $p_A^{V\circ U} \circ f_A^{V\circ U} = f_B^{V\circ U} = q_C^{V\circ U} \circ f_C^{V\circ U}$

(27) $f_A^{V\circ U} = p_B^{V\circ U} \circ q_C^{V\circ U} \circ f_C^{V\circ U}$ *from (24) and (26)*

(28) $f_C^{V\circ U} = q_B^{V\circ U} \circ p_A^{V\circ U} \circ f_A^{V\circ U}$ *from (25) and (26)*

Therefore we have $m_A^{V\circ U} = q_B^{V\circ U} \circ p_A^{V\circ U}$ and $m_C^{V\circ U} = p_B^{V\circ U} \circ q_C^{V\circ U}$ which suit the formalization in section 6.1. Hence, $V\circ U$ is conservative. Figure 6.4 shows the relevant objects and morphisms – $V\circ U^*$ and $h_B^{V\circ U}$ appear twice to improve readability.

Figure 6.4 – Diagram for conservativity in V-alignment composition.

$$\begin{tikzcd}
 & A^* & & & B^* & & & C^* & \\
A \arrow[ur, bend left, "j_A"] & & V^* \arrow[ul, "k_A^V"'] \arrow[urr, bend left, "k_B^V"] & & B \arrow[u, "j_B"] & & U^* \arrow[ull, bend right, "k_B^U"'] \arrow[ur, "k_C^U"] & & C \arrow[ul, bend right, "j_C"'] \\
 & & V\circ U^* \arrow[u, "k_V^{V\circ U}"] & & & & V\circ U^* \arrow[u, "k_U^{V\circ U}"] & & \\
 & A^V \arrow[uul, "g_A^V"] \arrow[uur, bend left, "h_A^V"] \arrow[rr, bend left, "m_A^V"] & & B^V \arrow[uul, bend right, "h_B^V"'] \arrow[uur, "g_B^V"] \arrow[ll, bend left, "m_B^V"] & & B^U \arrow[uul, "g_B^U"'] \arrow[uur, bend left, "h_B^U"] \arrow[rr, bend left, "m_B^U"] & & C^U \arrow[uul, bend right, "h_C^U"'] \arrow[uur, "g_C^U"'] \arrow[ll, bend left, "m_C^U"] & \\
 & A^{V\circ U} \arrow[u, "n_A"] \arrow[uur, bend left, "h_A^{V\circ U}"] \arrow[uuul, bend left, "g_A^{V\circ U}"] \arrow[rrr, bend left, "p_A^{V\circ U}"] & & & B^{V\circ U} \arrow[ul, bend right, "n_B^V"'] \arrow[ur, bend left, "n_B^U"] \arrow[uuu, "g_B^{V\circ U}"] \arrow[uull, bend left, "h_B^{V\circ U}"] \arrow[uurr, bend right, "h_B^{V\circ U}"'] \arrow[lll, bend left, "p_B^{V\circ U}"] \arrow[rrr, bend left, "q_B^{V\circ U}"] & & & C^{V\circ U} \arrow[u, "n_C"'] \arrow[uul, bend right, "h_C^{V\circ U}"'] \arrow[uuur, bend right, "g_C^{V\circ U}"'] \arrow[lll, bend left, "q_C^{V\circ U}"] & \\
 & & & & V\circ U \arrow[ulll, bend left, "f_A^{V\circ U}"] \arrow[u, "f_B^{V\circ U}"] \arrow[urrr, bend right, "f_C^{V\circ U}"'] & & & &
\end{tikzcd}$$

6.1.4 Section Summary

The findings presented in the previous subsections play an auxiliary role alongside our algorithm for correcting conservativity violations, presented in Chapter 7. They guide away from unnecessary executions of the algorithm over the result of an operation where the conservativity of both input alignments is known. Furthermore, the knowledge on the behavior of conservativity under V-alignment operations helps evaluating how useful it is to compute the operation in the first place.

Table 6.1 summarizes the findings discussed in the previous subsections. Given two V-alignments, V and U, each line in the table depicts a possible configuration regarding conservativity of V and U and the conservativity of the resulting alignment after the corresponding operation (i.e., intersection, union or composition). Conservativity is indicated by a check mark ("√"), non-conservativity by a cross mark ("✗") and uncertain by a question mark ("?").

Table 6.1 – Summary of the behavior of conservativity under V-alignment operations

Operation	*V conservative*	*U conservative*	*Result conservative*
Intersection	√	√	√
	√	✗	√
	✗	√	√
	✗	✗	?
Union	√	√	?
	√	✗	✗
	✗	√	✗
	✗	✗	✗
Composition	√	√	√
	√	✗	?
	✗	√	?
	✗	✗	?

Source: the author.

6.2 Conservativity in W-Alignments

Given two ontologies A and B, a W-alignment $(f_A{:}V_A \rightarrow A,\ f_W^A{:}V_A \rightarrow W,\ f_W^B{:}V_B \rightarrow W,\ f_B{:}V_B \rightarrow B)$ is non-conservative if merging both ontologies via the alignment (a colimit over the alignment, as discussed in Chapter 5) causes new subsumption relations to be introduced between two entities originated from the same source ontology. Due to the higher complexity of W-alignments, there are more points where conservativity violations may be introduced, beginning with the bridge ontology *W*. To check for conservativity violations introduced by the bridge ontology, this work treats each half of the alignment in a way that is similar to the treatment of V-alignments, however relaxed to allow the introduction of new subsumption relations between entities from the bridge ontology. This means that it is enough to check for the existence of a single morphism $m_A{:}W_V^A \rightarrow A_V^A$ (respectively, $m_B{:}W_V^B \rightarrow A_V^B$) for each half of the W-alignment.

Figure 6.5 – Diagram for conservativity in W-alignments.

$$\begin{tikzcd}
 & & A+ & & & & B+ & & \\
 & & & A+_W \arrow[ul, "g_{A+}^{W}"'] \arrow[rr, leftrightarrow, "m_{A+}\quad m_{B+}"] & & B+_W \arrow[ur, "g_{B+}^{W}"] & & & \\
A \arrow[uurr] & & & & W \arrow[ul, "f_{A+}"] \arrow[ur, "f_{B+}"'] & & & & B \arrow[uull] \\
 & A_V^A \arrow[ul, "g_A^V"'] & & W_V^A \arrow[ll, "m_A"'] \arrow[ur, "g_W^{VA}"] & & W_V^B \arrow[ul, "g_W^{VB}"'] \arrow[rr, "m_B"] & & B_V^B \arrow[ur, "g_B^V"] & \\
 & & V_A \arrow[ul, "f_A^V"] \arrow[ur, "f_W^{VA}"'] & & & & V_B \arrow[ul, "f_W^{VB}"] \arrow[ur, "f_B^V"'] & &
\end{tikzcd}$$

After checking for conservativity violations originating from the bridge ontology, it is necessary to verify if new subsumption relations are introduced by the aligned ontologies, taking into consideration the additional information from the bridge ontology. This verification requires the computation of the intermediate results *A+* and *B+*, which are the pushouts over (f_A, f_W^A) and (f_B, f_W^B) respectively, and the examination of the conservativity of the resulting V-alignment centered in W. Figure 6.5 depicts the full diagram for conservativity in W-alignments. It is worth of mention that computing *A+* and *B+* should be significantly easier and faster than computing the full merge, since it represents extending *A* and *B* with enough entities in order to cover the new information in *W*, which is usually considerably smaller than either ontology.

7 CORRECTING CONSERVATIVITY VIOLATIONS

According to the formalization of conservativity in V-alignments in Section 6.1, to verify if an alignment V is conservative it is first required to find the sub-ontologies A^V and B^V. Since the sub-ontologies contain every entity in the alignment and no other entity, their sets of entities are equal to the set of entities in the alignment, that is, $E_V = E_A{}^V = E_B{}^V$. More accurately, the functions $f_A{}^V$ and $f_B{}^V$, which map the entities in V to entities in A^V and B^V respectively, are bijective. Therefore, to find if there are morphisms $m_A{}^V{:}A^V \rightarrow B^V$ such that $m_A{}^V \circ f_A{}^V = f_B{}^V$ and $m_B{}^V{:}B^V \rightarrow A^V$ such that $m_B{}^V \circ f_B{}^V = f_A{}^V$, as required for conservative alignments, it is enough to check if there are no discrepancies between the subsumption relations in each sub-ontology.

Thus, for a non-conservative alignment, there are pairs (e_1, e_2) of entities in V such that $(e_1, e_2) \in S_A{}^V \oplus (e_1, e_2) \in S_B{}^V$, where $\oplus$ denotes exclusive logical disjunction. In order to correct the conservativity violations in V, it is necessary to remove from it a set of entities so that no such pair remains. Computing a vertex cover for a graph whose nodes are the entities and the edges are the discrepant subsumption relations results in a suitable set of entities to be removed for the alignment to become conservative.

It is desirable for the chosen repair, i.e., the set of entities to be removed from V, to be as small as possible. Finding the minimum vertex cover, however, is a NP-Complete problem (KARP, 1972), and there is no efficient algorithm to fulfill this task. Nonetheless, a simple greedy algorithm, described by Papadimitriou and Steiglitz (1998), provides an approximation that is no more than twice the size of the minimum vertex cover. The algorithm computes the cover by choosing edges from the graph, removing both endpoints from the graph and inserting them into the cover, until no remaining edges are left.

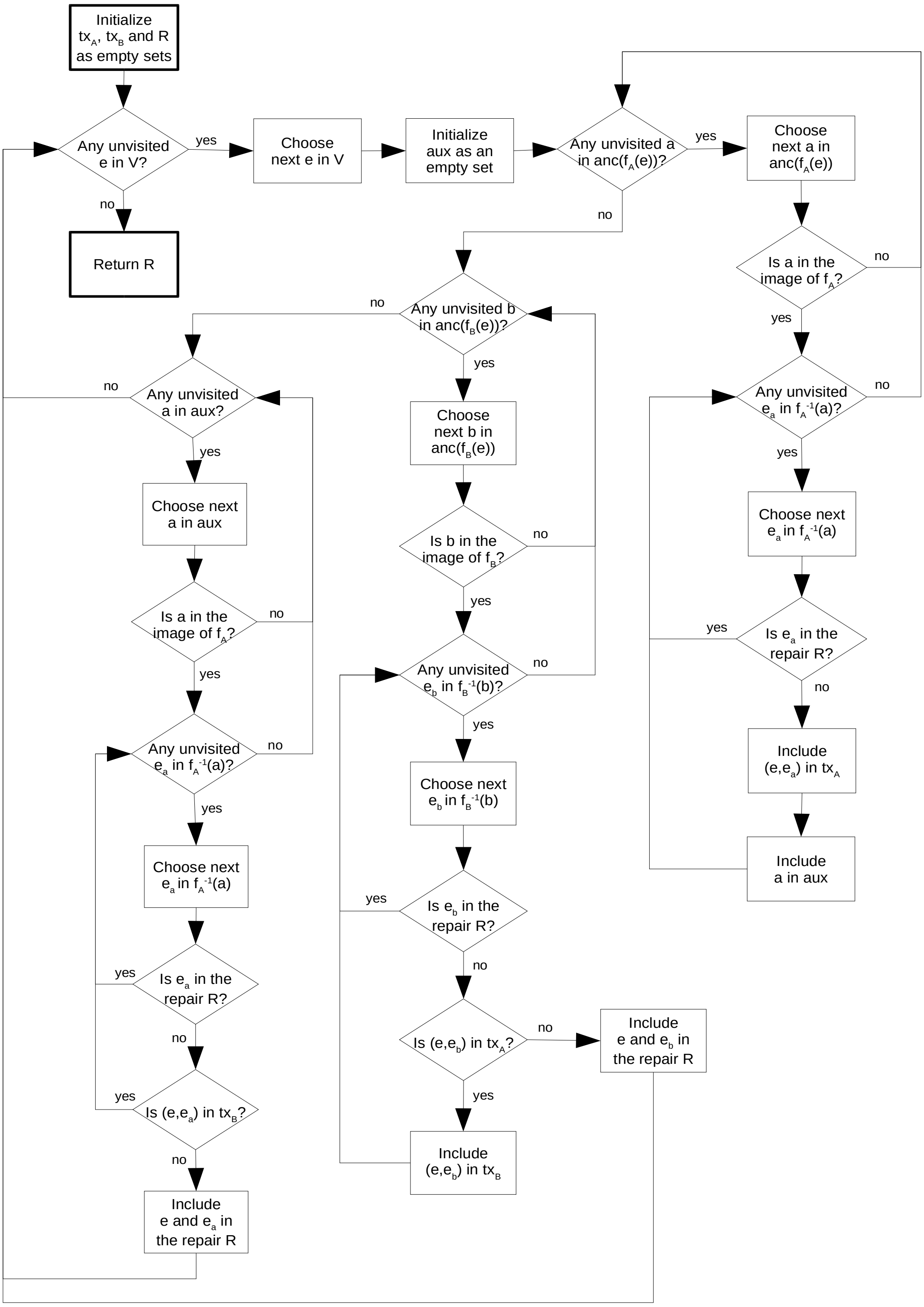
This work proposes an algorithm for correcting conservativity violations that is based on the greedy algorithm. Although there are slightly better approximations for the minimum vertex cover, such as the one proposed by Karakostas (2009), the proposed approach takes advantage of the greedy algorithm. Since the conservativity violation correction algorithm does not receive the graph as input, but must construct it from the alignment, it computes the cover at the same time as it is creating the graph. Whenever the graph construction process would introduce a new edge, both endpoints are included in the cover and removed from the graph.

Algorithm 1 presents the correction algorithm in detail, and Figure 7.1 displays the corresponding fluxogram. This algorithm represents the first of two steps in the correction pipeline. The second step, which this chapter shall discuss later, is an improvement phase which reduces the size of the computed repair.

Algorithm 1. Greedy algorithm for computing alignment repair

```
Input: $(V, f_A:V \to A, f_B:V \to B)$: alignment
       A, B: ontologies
Output: $R \subseteq V$: repair
    $tx_A \leftarrow \varnothing$
    $tx_B \leftarrow \varnothing$
    $R \leftarrow \varnothing$
L1: for each $e \in V$ do
        $aux \leftarrow \varnothing$
        for each $a \in ancestors(f_A(e)) \cap Im(f_A)$ do
            for each $e_a \in f_A^{-1}(a) - R$ do
                $tx_A \leftarrow tx_A \cup \{(e,e_a)\}$
                $aux \leftarrow aux \cup \{a\}$
        for each $b \in ancestors(f_B(e)) \cap Im(f_B)$ do
            for each $e_b \in f_B^{-1}(b) - R$ do
                if $(e,e_b) \notin tx_A$ do
                    $R \leftarrow R \cup \{e, e_b\}$
                    go to next iteration of L1
                else do
                    $tx_B \leftarrow tx_B \cup \{(e,e_b)\}$
        for each $a \in aux \cap Im(f_A)$ do
            for each $e_a \in f_A^{-1}(a) - R$ do
                if $(e,e_a) \notin tx_B$ do
                    $R \leftarrow R \cup \{e, e_a\}$
                    go to next iteration of L1
    return $R, tx_A, tx_B$
```

Figure 7.1 – Fluxogram for repair algorithm.



Source: the author.

Algorithm 1 receives as input a V-alignment $(V, f_A{:}V \rightarrow A, f_B{:}V \rightarrow B)$ and the concept hierarchies for both aligned ontologies provided by an OWL 2 reasoner. It initializes the repair as an empty set, and constructs the taxonomies tx_A and tx_B, also initialized as empty sets, of subsumption relations between the aligned entities in ontology *A* and *B* respectively. The algorithm iterates over the entities in *V* (line 4), checking the ancestors of their corresponding entities in the aligned ontologies. Whenever it finds a subsumption relation in *A* (lines 6 and 7), the algorithm includes it in tx_A and in an auxiliary set for later comparison with the subsumption relations in *B* (lines 8 and 9). Then, it checks the subsumption relations in *B* (lines 10 and 11), include them in tx_B and compares to tx_A (line 12). If there is no corresponding relation in tx_A, the algorithm includes both involved entities in the repair (that is, remove them from the alignment – line 13). Finally, it iterates over the entities in the auxiliary set and compare the corresponding subsumption relations to those in tx_B, inserting offenders in the repair (lines 19 and 20). Whenever the algorithm finds a violation and insert entities in the repair, it proceeds on to the next entity in the alignment. Once there are no remaining entities to analyze, the algorithm finishes and returns the computed repair.

This method is enough to cover violations of equivalence conservativity in addition to violations of subsumption conservativity. If two entities *e* and *d* in *V* are mapped to a single entity in *A* (i.e., $f_A(e) = f_A(d)$) and to different entities in *B*, the subsumption relations (e, d) and (d, e) will be present in tx_A. This happens because subsumption is reflexive, that is, every entity subsumes itself, and therefore $f_A(e)$ *subsumes* $f_A(d)$ and vice versa. The corresponding subsumption relations will not be present in tx_B, however, unless f_B maps *e* and *d* to equivalent entities in *B*. The algorithm thus detects the introduction of a new equivalence relation in *B*. If, on the other hand, the equivalence relation results from a circular chain of subsumption relations, the algorithm will detect the new subsumption relations which are responsible for the chain.

An attentive reader may notice that the algorithm deals only with the ancestors of the entities in the alignment. It is not necessary to replicate the steps conducted for the descendants of these entities because the algorithm cares only for subsumption relations between entities in the alignment and it iterates over all such entities. Therefore, even if the algorithm may “ignore” the descendants of an entity when it iterates over that entity, the algorithm will select in future iterations those descendants which are in the alignment and find the original entity as an ancestor. Thus, the algorithm will examine the corresponding subsumption relation. The same would be true if the algorithm cared only for descendants, and the choice between iterating over ancestors or descendants is largely arbitrary.

Since the algorithm is based on the greedy algorithm for finding a vertex cover (PAPADIMITRIOU; STEIGLITZ, 1998), the resulting repair is an approximation of the minimum possible repair which may be up to twice its size. This may lead to very aggressive repair strategies that remove a large amount of information from the alignment. In order to reduce the repair's impact over the alignment, this work proposes a subsequent improvement step over the resulting repair.

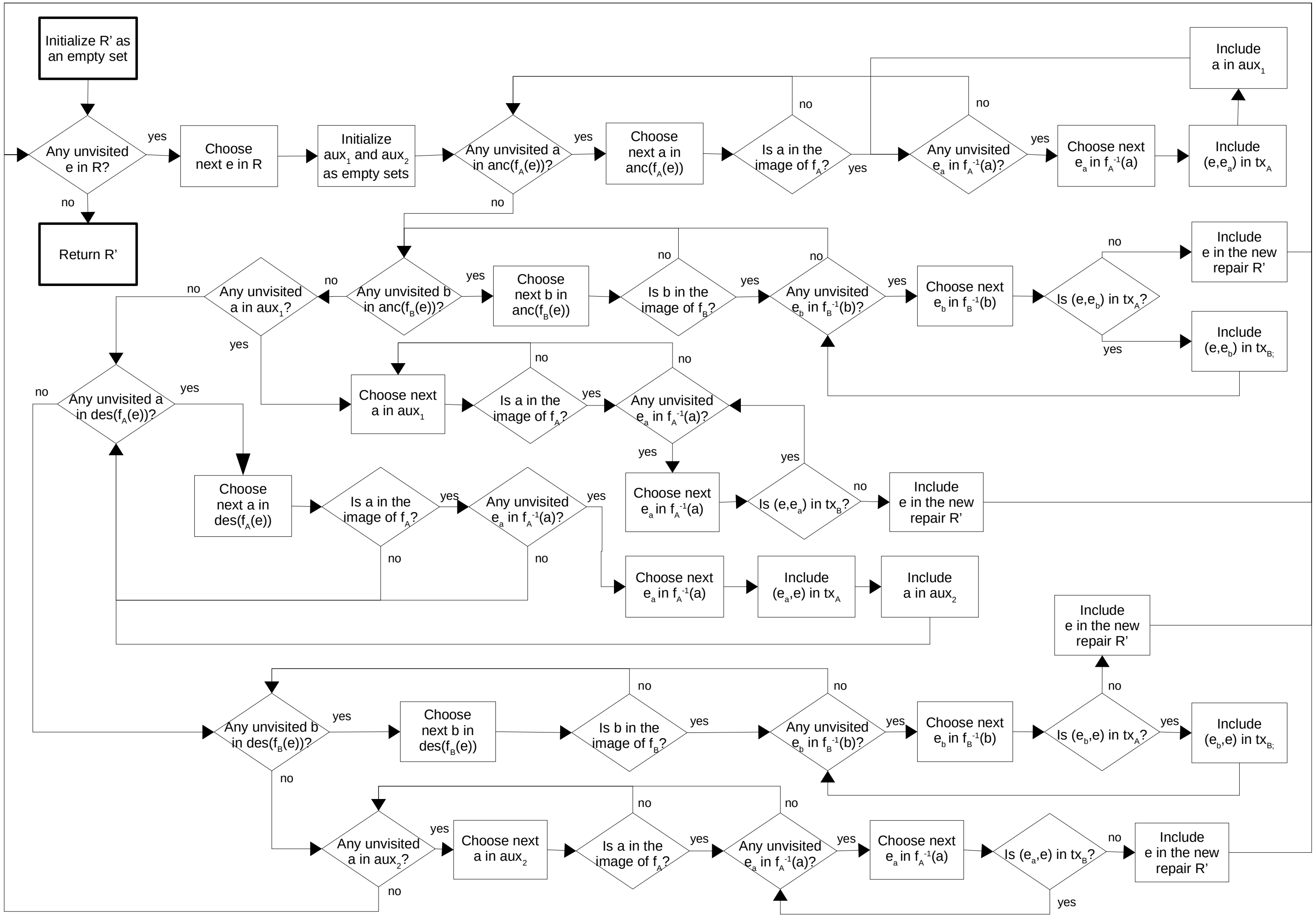
Algorithm 2 depicts the improvement step. The algorithm iterates over the entities in the original repair and verifies if reinserting them one by one in the alignment would reintroduce conservativity violations. The algorithm includes those entities which do cause conservativity violations in the new repair, and inserts those which do not back in the alignment. Thus, while Algorithm 1 includes two entities in the repair at a time, Algorithm 2 includes only one.

Differently from Algorithm 1, Algorithm 2 does not iterate over the whole alignment and therefore must check both the ancestors and the descendants of each entity to find the discrepant subsumption relations between the aligned ontologies. Thus, the loops appear "repeated" for iterating over the descendants. Apart from these key differences, both algorithms are very similar. It is also noteworthy that, for every edge that share no endpoint with any other edge in the graph, Algorithm 1 removes both endpoints from the alignment, while Algorithm 2 removes only one. Therefore, in the worst-case scenario for Algorithm 1's approximation, that is, when the size of the computed repair is twice the size of the minimum repair, where no edges share nodes, Algorithm 2 reduces the repair size in half – i.e., to the minimum repair. In the practical evaluation of the algorithms in Chapter 7, this was the case for all alignments in the Conference dataset. Although the reduction will not be always so significant, the resulting repair will always be smaller than twice the size of the minimum.

Algorithm 2. Repair improvement algorithm

```
Input: $(V, f_A:V \to A, f_B:V \to B)$: alignment
       $R \subseteq V$: repair
       $tx_A$, $tx_B$: taxonomies
Output: $R' \subseteq R$: improved repair
    $R' \leftarrow \varnothing$
L1: for each $e \in R$ do
        $aux_1 \leftarrow \varnothing$
        for each $a \in ancestors(f_A(e)) \cap Im(f_A)$ do
            for each $e_a \in f_A^{-1}(a)$ do
                $tx_A \leftarrow tx_A \cup \{(e,e_a)\}$
                $aux_1 \leftarrow aux_1 \cup \{a\}$
        $aux_2 \leftarrow \varnothing$
        for each $d \in descendants(f_A(e)) \cap Im(f_A)$ do
            for each $e_d \in f_A^{-1}(d)$ do
                $tx_A \leftarrow tx_A \cup \{(e_d,e)\}$
                $aux_2 \leftarrow aux_2 \cup \{d\}$
        for each $b \in ancestors(f_B(e)) \cap Im(f_B)$ do
            for each $e_b \in f_B^{-1}(b)$ do
                if $(e,e_b) \notin tx_A$ do
                    $R' \leftarrow R' \cup \{e\}$
                    go to next iteration of L1
                else do
                    $tx_B \leftarrow tx_B \cup \{(e,e_b)\}$
        for each $c \in descendants(f_B(e)) \cap Im(f_B)$ do
            for each $e_c \in f_B^{-1}(c)$ do
                if $(e_c,e) \notin tx_A$ do
                    $R' \leftarrow R' \cup \{e\}$
                    go to next iteration of L1
                else do
                    $tx_B \leftarrow tx_B \cup \{(e_c,e)\}$
        for each $a \in aux_1 \cap Im(f_A)$ do
            for each $e_a \in f_A^{-1}(a)$ do
                if $(e,e_a) \notin tx_B$ do
                    $R' \leftarrow R' \cup \{e\}$
                    go to next iteration of L1
        for each $d \in aux_2 \cap Im(f_A)$ do
            for each $e_d \in f_A^{-1}(d)$ do
                if $(e_d,e) \notin tx_B$ do
                    $R' \leftarrow R' \cup \{e\}$
                    go to next iteration of L1
    return $R'$
```

Figure 7.2 – Fluxogram for repair improvement algorithm



Source: the author.

Finally, Algorithm 3 pieces algorithms 1 and 2 together and depicts our full approach for correcting conservativity violations.

Algorithm 3. Full correction algorithm

```
Input: $(V, f_A{:}V \to A, f_B{:}V \to B)$: alignment
       A, B: ontologies
Output: R: alignment repair
    R, $tx_A$, $tx_B$ ← Alg1($(V,f_A,f_B)$,A,B)
    R ← Alg2($(V,f_A,f_B)$,R,$tx_A$,$tx_B$)
    return R
```

7.1 Time complexity analysis

Looking at Algorithm 1, the first three lines are all constant time operations. Line 4 begins a loop that ends in line 23 and iterates over all the *n* entities in the alignment. The loop contains one constant-time operation in line 5, followed by three other loops beginning in lines 6, 10 and 16. The first and third loops iterate over ancestors of the selected aligned entity in ontology A, and the second loop iterates over ancestors in ontology B. Each loop contains an inner loop that iterates over the entities in the alignment which are mapped to the selected ancestor. Since each entity in the alignment may not be mapped to more than one entity in each ontology, the total number of executions of lines 8 through 9, 12 through 15 and 18 through 23 must not be greater than the number *n* of entities in the alignment. This reflects the fact that the total sum of the size of the sets of entities in the alignment which are mapped to each ancestor of any entity must not be greater than the size of the alignment, precisely because those sets are disjoint.

Proposition 7.1. The average time complexity of Algorithm 1 is $O(n^2)$.

Proof. With the proper choice of data structures, such as Python's dictionaries (which have an average time complexity of O(1) for insertions and also for checking if an element is in the dictionary) (HARTLEY, 2017), all operations in lines 8 through 9, 12 through 15 and 18 through 23 may be computed in constant time. Therefore, the upper bound for the average time complexity of each of the loops in lines 6, 10 and 16 is $O(n)$. Since ontologies are usually not densely connected and alignments tend not to map several entities to a single one, the actual number of iterations is frequently much smaller. The average time complexity of the whole algorithm is, consequently, asymptotically bounded by $O(n^2)$.

Proposition 7.2. The average time complexity of Algorithm 2 is $O(n^2)$.

Proof. Algorithm 2 has a similar structure to Algorithm 1, only with a duplication of the loops in order to account for the entity’s descendants as well as its ancestors. Since the new loops are included sequentially and present the same complexity limits, the asymptotic bound for the resulting time complexity is not changed.

8 EXPERIMENTAL EVALUATION

The proposed algorithm was evaluated over the reference alignments from three tracks of the Ontology Alignment Evaluation Initiative (OAEI) 2019[6]. The Conference track provides 21 manually created alignments between seven ontologies from the OntoFarm collection (ZAMAZAL; SVÁTEK, 2017), covering the domain of academic conferences. This dataset contains small ontologies (the biggest has 141 classes) and small alignments (up to 26 mappings). The Anatomy track comprises the Adult Mouse Anatomy (AMA) ontology and a part of the National Cancer Institute (NCI) Thesaurus describing human anatomy (ZHANG; MORK; BODENREIDER, 2004), along with a reference alignment containing 1516 mappings. Finally, the LargeBio track involves three ontologies with tens of thousands of classes each, the Foundational Model of Anatomy (FMA), the Systematized Nomenclature of Medicine (SNOMED) – Clinical Terms and the NCI Thesaurus. The track also contains three reference alignments based on the Unified Medical Language System (UMLS) Metathesaurus (BODENREIDER, 2004) and built by a combination of automatic techniques, expert assessment, and auditing protocols.

In the evaluation, we assess mainly two aspects: the execution time and the size of the repair. In both cases, the smaller the value the better. A shorter execution time reflects higher efficiency, and a smaller repair size means that less information will be removed from the alignment.

The algorithm was implemented in Python 3[7] with the libraries RDFLib and Owlready2 (LAMY, 2017), the latter of which includes a modified version of the HermiT reasoner. The evaluations were performed on an Ubuntu 18.10 desktop computer with an Intel Core i7-7700 CPU @ 3.60GHz × 8 and 16 GB of RAM. Table 8.1 presents the results. The column “input size” displays the size of the original alignment, the columns “Alg. 1 time (s)” and “Alg. 1 repair size” present the execution time in seconds of the greedy algorithm and the size of the resulting repair, i.e., the number of modifications over the alignment. Algorithm 2 then receives this repair as input, and the columns “Alg. 2 time (s)” and “Alg. 2 repair size” display its execution time and the final repair size. Finally, the percentual repair size refers to the ratio of mappings from the input alignment that were altered (in the present case, removed) by the final repair after the improvement step. This last column indicates the actual

---

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

[7] The source code and corrected alignments are available at
http://github.com/BDI-UFRGS/conservativity/

impact of the computed repair over the alignment. The most aggressive repair computed removed 42.4% of the mappings from the SNOMED-NCI alignment in the LargeBio dataset.

Table 8.1 – Results of the experimental evaluation over reference alignments from OAEI 2019

Dataset	*Ontologies*	*Input size*	*Alg. 1 repair size*	*Alg. 2 repair size*	*Alg. 1 time (s)*	*Alg. 2 time (s)*	*Repair size (%)*
Conference	cmt-conference	15	2	1	.00034	.00008	6.7
	cmt-confof	16	4	2	.00035	.00023	12.5
	cmt-ekaw	11	2	1	.00024	.00020	9.1
	cmt-sigkdd	12	2	1	.00030	.00010	8.3
	conference-edas	17	2	1	.00038	.00013	5.9
	conference-ekaw	25	2	1	.00057	.00032	4.0
	conference-iasted	14	2	1	.00034	.00013	7.1
	confof-ekaw	20	4	2	.00048	.00016	10.0
	edas-ekaw	23	4	2	.00049	.00021	8.7
	edas-iasted	19	4	2	.00045	.00028	10.5
Anatomy	AMA-NCI	1516	338	181	.04307	.03713	11.9
LargeBio	FMA-NCI	3024	1376	800	.11024	.13855	26.5
	FMA-SNOMED	9008	5218	3244	.73732	1.85429	36.0
	SNOMED-NCI	18844	12028	7995	1.43010	8.73326	42.4

Source: the author.

It is noteworthy that the improvement step (i.e., Algorithm 2) reduced the repair resulting from the greedy algorithm to half its size for all alignments from the Conference dataset. As previously discussed, this means that the algorithm has reached the minimum possible repair that removes the conservativity violations from the alignment. The reduction rate provided by Algorithm 2 ranged from 33.5% (again for the SNOMED-NCI alignment) to the maximum possible reduction of 50% in the Conference dataset.

The execution time of both steps of the algorithm was small – even for the largest alignment, with over 18 thousand mappings, the combined total was of 10.16 seconds. The most significant bottleneck in the entire correction process was actually the execution of the OWL 2 reasoner, which is outside the scope of the current work but is a prerequisite for the algorithm to remove all possible conservativity violations. The use of module extraction techniques to select only the significant portion of the input ontologies prior to running the reasoner may go a long way towards reducing said bottleneck, but the modularization approach must be chosen carefully, since modules of poor quality will reflect directly on conservativity violations persisting in the resulting alignment.

These results were compared to the performance of the LogMap[8] conservativity extension designed by Solimando, Jiménez-Ruiz and Guerrini (2014a, 2014b, 2017), referred to in the remainder of this dissertation as LogMapC. Table 8.2 outline the results of the tests, which were executed with the same inputs and in the same computational context. Besides removing mappings from the alignment, LogMapC also replace some equivalence mappings by subsumption mappings. The total repair size is simply the sum of the number of mappings that were removed and the number of equivalence mappings replaced with subsumption mappings.

Table 8.2 – Results of the evaluation of LogMapC over reference alignments from OAEI 2019

Dataset	*Ontologies*	*Input size*	*Repair size*	*Removed mappings*	*Replaced mappings*	*LogMap time (s)*	*Repair size (%)*
Conference	cmt-conference	15	4	4	0	.056	26.7
	cmt-confof	16	6	6	0	.039	37.5
	cmt-ekaw	11	1	1	0	.036	9.1
	cmt-sigkdd	12	1	0	1	.025	8.3
	conference-edas	17	3	1	2	.062	17.6
	conference-ekaw	25	2	1	1	.029	8.0
	conference-iasted	14	1	0	1	.042	7.1
	confof-ekaw	20	2	2	0	.032	10.0
	edas-ekaw	23	4	0	4	.073	17.4
	edas-iasted	19	4	4	0	.069	21.1
Anatomy	AMA-NCI	1516	302	132	170	1.372	19.9
LargeBio	FMA-NCI	3024	1744	934	810	5.998	57.7
	FMA-SNOMED	9008	8387	3864	4523	192.840	93.1
	SNOMED-NCI	18844	12961	6679	6282	2167.728	68.8

Source: the author.

In the case of LogMapC, the percentage of mappings altered (i.e., either removed or replaced with subsumption mappings) ranged from 8% to 93.1%. The later is verified for the FMA-SNOMED alignment in the LargeBio dataset, where 8387 mappings where removed or replaced out of the original 9008. The execution time, which again does not include the time required for computing modules from the source ontologies and running the reasoner, reached over 36 minutes for the SNOMED-NCI alignment.

The comparison between the two algorithms is summarized in Table 8.3. Each column presents the percentual difference between the size or the time obtained by LogMapC and the

---

[8] LogMapC was compiled from the source code available at http://github.com/ernestojimenezruiz/logmap-conservativity/

respective value obtained by the algorithm presented in the current work. The higher the percentage the greater the improvement provided by our algorithm. For example, the number of modifications in the repair computed by the proposed algorithm for the FMA-SNOMED alignment was 61.3% smaller than the number of modifications computed by LogMapC for the same alignment if both mapping removal or its replacement are counted as single modifications. If, however, the removal of an equivalence mapping is counted twice, i.e., as the removal of two subsumption mappings, as conducted by Solimando, Jiménez-Ruiz and Guerrini (2014a, 2014b, 2017), the repair was only 47% less aggressive.

Table 8.3 – Comparison between the proposed algorithm and LogMapC

Dataset	*Ontologies*	*Input size*	*Repair size difference (%)*[9]	*Repair size difference (%)*[10]	*Execution time improvement (%)*
Conference	cmt-conference	15	75.0	75.0	99.3
	cmt-confof	16	66.7	66.7	98.5
	cmt-ekaw	11	0.0	0.0	98.8
	cmt-sigkdd	12	0.0	-100.0	98.4
	conference-edas	17	66.7	50.0	99.2
	conference-ekaw	25	50.0	33.3	96.9
	conference-iasted	14	0.0	-100.0	98.9
	confof-ekaw	20	0.0	0.0	98.0
	edas-ekaw	23	50.0	0.0	99.0
	edas-iasted	19	50.0	50.0	98.9
Anatomy	AMA-NCI	1516	40.1	16.6	94.2
LargeBio	FMA-NCI	3024	54.1	40.3	95.9
	FMA-SNOMED	9008	61.3	47.0	98.7
	SNOMED-NCI	18844	38.3	18.6	99.5

Source: the author.

The proposed algorithm compares very favorably to LogMapC. Even when counting the removal of equivalence mappings as two modifications – which is favorable to LogMapC, since it is the only algorithm that performs other types of modifications – there where only two cases where LogMapC's repair was smaller: for the alignments cmt-sigkdd and conference-iasted in the Conference dataset, both cases where the proposed algorithm's repair

---

[9] Counting both removing an equivalence mapping or replacing it with a subsumption mapping as a single modification

[10] Counting removing an equivalence mapping as removing two subsumption mappings (i.e., as two modifications)

contains a single removal and the repair computed by LogMapC contains the replacement of a single equivalence mapping for a subsumption mapping.

In other two cases, both algorithms recommended the removal of the same number of mappings, for the alignments cmt-ekaw and confof-ekaw. Finally, for the alignment edas-ekaw, LogMapC recommends the substitution of two equivalence mappings for subsumption mappings, while the algorithm proposed here recommends the removal of a single equivalence mapping – since removing an equivalence mapping counts as two modifications, both repairs are deemed as equally aggressive. These three cases are also in the Conference dataset. For every other alignment, the repair computed by the proposed algorithm is considerably less aggressive, ranging from 16.6% to 75% smaller.

When it comes to the execution time, LogMapC was over 10 times slower for every input case. In the case of the SNOMED-NCI alignment, which was the most time-intensive for both algorithms, the execution time for the proposed algorithm was only 0.5% of the time spent by LogMapC.

A final aspect to be considered is that LogMapC does not guarantee that the resulting alignment is completely free of conservativity violations. In fact, for the alignments AMA-NCI from the Anatomy track and FMA-SNOMED and SNOMED-NCI from the LargeBio track, the alignments still contained 4, 56 and 528 violations, respectively, after the repair computed by LogMapC. Meanwhile, for the alignments corrected by the algorithm proposed here, zero violations were detected by LogMapC except for the SNOMED-NCI alignment, in which 1227 remaining violations were detected. However, none of these violations were found in the resulting merged ontology, and are therefore false positives in the result of LogMapC.

9 CONCLUSION

In the context of the Semantic Web, targeted at computer programs rather than human users, the validation and correction of ontology alignments takes a prominent role in the actual realization of semantic interoperability. Further, efficiency in evaluating alignments obtained from diverse sources on the web is key in enabling the automatic processing of decentralized information.

This dissertation has provided a category-theoretic formalization for conservativity in ontology alignments. The foundation in category theory allowed the analysis of conservativity under alignment operations formalized with the same mathematical formalism. Particularly, this work has presented the proof that the intersection of a conservative V-alignment with any other V-alignment is also conservative, that the union of a non-conservative V-alignment with any other V-alignment is also non-conservative and that the composition of two conservative V-alignments is always conservative. These proofs guide away from unnecessary verifications of the conservativity of the result of an operation where the conservativity of the input alignments is known, and also help evaluating the usefulness of such operations when it comes to finding conservative alignments.

Additionally, the formalization served as a basis for the development of a two-step approach for automatically correcting conservativity violations in ontology alignments. The first step consists of a greedy algorithm that removes from the alignment pairs of entities between which a subsumption relation is introduced until no such pairs remain. The second step is an improvement step, aimed at reducing the impact of the repair process over the original alignment, and reintroduces in the alignment some of the entities removed in the first step, as long as the reintroduction does not lead to new conservativity violations.

The algorithm was designed to correct both violations of subsumption conservativity and violations of equivalence conservativity, without requiring the previous merging of the aligned ontologies and the execution of reasoners over the merged ontology – an advantage over the approaches of Meilicke (2006), Jean-Marie, Shinoroshita and Kabuka (2009), Lambrix and Liu (2013) and Ivanova and Lambrix (2013) – neither the assumption of disjointness and corresponding insertion of additional disjointness axioms, both of which lead to great complexity and a large expenditure of time when the input ontologies are large.

An experimental comparison between the proposed approach and the conservativity extension of LogMap developed by Solimando, Jiménez-Ruiz and Guerrini (2014a, 2014b, 2017) shows that the proposed two-steps algorithm is significantly more efficient, ranging from 10 to 200 times faster, and in most cases results in less aggressive repairs that nevertheless remove all violations from the input alignments. Therefore, the repairs computed by the proposed algorithm preserve more information from the input alignment.

Nonetheless, the proposed algorithm still has room for improvement. As it stands now, the algorithms select entities in any order, largely dependent on the actual implementation of the data structures, which ultimately affect the computed repair – albeit in experimental testing the difference in size was not significant. The use of some heuristics for the order of selection of nodes might contribute towards further reducing the impact of the repair in the input alignment. Another refinement that could promote repairs that are even less aggressive is the inclusion of an additional step to check if the removed equivalence mappings may be reintroduced in the alignment as subsumption mappings without introducing new violations. Also, the addition of a module-extraction step prior to submitting the ontologies to a reasoner may contribute to decrease the most significant bottleneck in the actual execution of the algorithm.

Another path forward is the adaptation of the algorithm for W-alignments. In the case of the datasets from the Ontology Alignment Initiative 2019, all alignments contained only equivalence mappings and could be represented as V-alignments. Previous editions of OAEI, however, had alignments that matched some concepts by subsumption rather than by equivalence. These alignments can only be represented as W-alignments. In order to update the algorithm to correct W-alignments, it is required to first make a “directed” version of the algorithm in order to verify, for each leg of the W, if new subsumption relations are being introduced by the bridge ontology in the aligned ontology – new subsumption relations in the bridge ontology are not an issue and are actually to be expected. After performing this correction separately over each leg of the W, the aligned ontologies must be extended with the entities and relations in the bridge ontology. Then, the algorithm may be used as presented in this dissertation taking the bridge ontology as an alignment between the extended ontologies to correct conservativity violations introduced in one aligned ontology by the other. Finally, any changes made to the bridge ontology must then be propagated to the alignments in each leg of the W.

APPENDIX RESUMO EM PORTUGUÊS

Diferente da rede mundial de computadores tradicional, que é voltada para usuários humanos, a Web Semântica é feita para ser acessada por programas de computador. Para permitir isso, o significado dos dados deve ser especificado formalmente em ontologias, artefatos computacionais que descrevem explicitamente alguma porção da realidade de acordo com a visão de certa comunidade.

É frequente que diversas comunidades produzam ontologias diferentes para lidar com o mesmo domínio ou com domínios relacionados. Nestes cassos, é útil combinar o conhecimento das diferentes ontologias. Essa integração é particularmente importante em contextos distribuídos como a Web Semântica, onde a informação pode vir de muitas fontes distintas. Para integrar duas ontologias, é necessário descrever mapeamentos entre conceitos de ambas ontologias através de um alinhamento.

Ainda que existam diversas técnicas para alinhar ontologias, tanto manuais quanto automaticas, nenhuma é livre de erros. Alinhamentos imperfeitos entre ontologias usualmente levam a consequências não pretendidas. Entre as consequências mais comuns de alinhamentos falhos é a violação do princípio de conservatividade.

O princípio de conservatividade estabelece que a união de duas ontologias a partir de um alinhamento não deve introduzir nenhuma nova relação de subsunção ou de equivalência entre conceitos que se originam na mesma ontologia. Uma relação de subsunção existe entre dois conceitos *c* e *d* se, e somente se, todas as instâncias de *d* são também instâncias de *c*, e é dito que *c subsume d*. Uma relação de equivalência existe entre dois conceitos se eles compartilham todas as instâncias em todos as situações possíveis. Isto é, *c é equivalente a d* se, e somente se, todas as instâncias de *c* são instâncias de *d* e toda instância de *d* é uma instância de *c*. Dessa forma, equivalência implica e é implicada por subsunção mútua.

Sempre que o princípio de conservatividade é violado, a ontologia resultante não preserva o significado pretendido dos conceitos especificados nas ontologias originais. Tomando como exemplo um alinhamento que mapeia os conceitos *pessoa*, *cliente* e *empresa* das ontologias *A* e *B*, onde a ontologia *A* especifica que *pessoa* subsume *cliente* (ou seja, todo cliente é uma pessoa) e a ontologia *B* define que *cliente* subsume *empresa* (toda empresa é um cliente). Se o alinhamento for usado para unir as duas ontologias, a ontologia resultante especificará que *pessoa* subsume *empresa*, isto é, todas empresas são pessoas, o que não reflete a especificação de nenhuma das duas ontologias originais. Essa discordância reflete que a conceituação expressa na ontologia resultante é incompatível com aquelas das ontologias de entrada.

Essa incompatibilidade indica que o alinhamento está mapeando conceitos cujos significados são diferentes, isto é, está igualando conceitos que não são os mesmos. Estes mapeamentos podem ocorrer por similaridades terminológicas ou estruturais entre os conceitos, que levam o algoritmo de mapeamento ou o especialista de domínio a acreditar que os seus significados são equivalentes, ainda que na realidade eles não sejam. O objetivo do presente trabalho é melhorar alinhamentos de ontologias removendo mapeamentos incorretos que se originam dessas falsas equivalências. As violações do princípio de conservatividade são tomadas como evidências de mapeamentos incorretos.

No exemplo anterior, o conceito *cliente* na ontologia *A* se refere especificamente a clientes humanos, enquanto na ontologia *B* o conceito se refere a uma gama maior de entidades, incluindo clientes corporativos, isto é, empresas que são clientes de outras empresas. Ainda, na ontologia *B* o conceito *empresa* na verdade se refere somente a clientes corporativos, excluindo fornecedores, por exemplo, enquanto o conceito *empresa* na ontologia *A* não tem essa restrição. Dessa forma, os conceitos não tem os mesmos significados nas duas ontologias. A introdução de novas relações de subsunção pelo alinhamento, violando o princípio da conservatividade, envidencia este fato.

No contexto da Web Semântica, a validação e correção de alinhamentos de ontologias tem um importante papel na realização prática da interoperabilidade semântica. Além disso, a eficiência na avaliação de alinhamentos obtidos de diversas fontes é uma peça chave em possibilitar o processamento automático de informações descentralizadas.

A presente dissertação apresenta uma formalização em teoria das categorias para alinhamentos de ontologias. A fundamentação em teoria das categorias permitiu a análise da conservatividade sob operações de alinhamentos definidas com o mesmo formalismo. Particularmente, este trabalho apresenta provas de que a intersecção de um alinhamento conservativo com qualquer outro alinhamento resulta em um alinhamento conservativo, assim como de que a união de um alinhamento não-conservativo com qualquer outro alinhamento resulta em um alinhamento não conservativo, e que a composição de dois alinhamentos conservativos é sempre conservativa. Essas provas auxiliam a evitar verificações desnecessárias da conservatividade do resultado de uma operação onde a conservatividade dos dois alinhamentos de entrada é conhecida, e também a avaliar a utilidade destas operações em primeiro lugar.

Adicionalmente, a formalização em teoria das categorias serviu como base para o desenvolvimento de uma técnica de duas etapas para a correção automatica de violações de conservatividade em alinhamentos de ontologias. A primeira etapa consiste em um algoritmo guloso que remove do alinhamento pares de entidades entre as quais uma relação de subsunção seria introduzida, até que nenhum par do tipo permaneça. A segunda etapa é uma etapa de melhoria, voltada a reduzir o impacto do processo de reparo sobre o alinhamento original, que reintroduz no alinhamento algumas das entidades removidas na primeira etapa, desde que isso não leve à reintrodução de violações de conservatividade.

O algoritmo foi desenvolvido para corrigir as violações de conservatividade sem a necessitade de fundir previamente as duas ontologies e executar raciocinadores sobre a ontologia resultante – uma vantagem sobre as propostas de Meilicke (2006), Jean-Marie, Shinoroshita e Kabuka (2009), Lambrix e Liu (2013) e de Ivanova e Lambrix (2013) – e sem a necessidade da suposição de que os conceitos da ontologia são disjuntos e da correspondente inserção de axiomas de disjunção adicionais – uma vantagem em relação à proposta de Solimando, Jiménez-Ruis e Guerrini (2014a, 2014b, 2017) – ambas tarefas que levam a grande complexidade e a um grande dispêndio de tempo quando as ontologias de entrada são grandes.

Uma comparação prática entre a técnica apresentada nesta dissertação e o algoritmo desenvolvido por Solimando, Jiménez-Ruis e Guerrini (2014a, 2014b, 2017) mostra que o algoritmo proposto é significativamente mais eficiente, com execuções de 10 a 200 vezes mais rápidas, e na maioria dos casos resultando em reparos menos agressivos que ainda assim removem todas as violações dos alinhamentos de entrada. Dessa forma, os reparos computados pelo algoritmo proposto preservam mais informações vindas do alinhamento de entrada.

REFERENCES

ADÁMEK, J.; HERRLICH, H.; STRECKER, G. E. **Abstract and concrete categories**: The joy of cats. New York: John Wiley and Sons, Inc, 1990.

ANTONIOU, G.; VAN HARMELEN, F. Web Ontology Language: OWL. In: **Handbook on Ontologies**. Heidelberg: Springer, 2009. p. 91–110.

ANTUNES, C; RADEMAKER, A.; ABEL, M. A category-theoretic approach for the detection of conservativity violations in ontology alignments. **Seminar on Ontology Research in Brazil**. Porto Alegre, 2019.

ANTUNES, C; RADEMAKER, A.; ABEL, M. A faster and less aggressive algorithm for correcting conservativity violations in ontology alignments. In: **Applied Ontology**, in press.

BENCH-CAPON, T.; MALCOLM, G. Formalising ontologies and their relations. **International Conference on Database and Expert Systems Applications**. Florence, 1999.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. In: **Scientific American**, New York, v. 284, n. 5, p. 34-43, 2001.

BODENREIDER, O. The Unified Medical Language System (UMLS): integrating biomedical terminology. In: **Nucleic Acids Research**, Oxford, v.32, p. 267-270.

CAFEZEIRO, I.; HAEUSLER, E. H. Semantic interoperability via category theory. **International Conference on Conceptual Modeling**. Auckland, 2007.

GUARINO, N.; OBERLE, D.; STAAB, S. What is an ontology? In: **Handbook on ontologies**. Heidelberg: Springer, 2009. p. 1–17.

HARTLEY, J. **Time Complexity in Python**. 2017. Available at: https://wiki.python.org/moin/TimeComplexity. Accessed 12 April 2020.

IVANOVA, V.; LAMBRIX, P. A unified approach for aligning taxonomies and debugging taxonomies and their alignments. **Extended Semantic Web Conference**. 2013.

JEAN-MARIE, Y. R.; SHINOROSHITA, E. P.; KABUKA, M. R. Ontology matching with semantic verification. In: **Journal of Web Semantics**, [s.l.], v. 7, p. 232-251.

JIMÉNEZ-RUIZ, E.; GRAU, B. C. LogMap: logic-based and scalable ontology matching. **International Semantic Web Conference**. 2011.

JIMÉNEZ-RUIZ, E.; GRAU, B. C.; HORROCKS, I.; BERLANGA, R. Logic-based assessment of the compatibility of UMLS ontology sources. In: **Journal of Biomedical Semantics**, v. 2, n. 1, p. S2, 2011.

KARAKOSTAS, G. A better approximation ratio for the vertex cover problem. In: CAIRES, L.; ITALIANO, G. F.; MONTEIRO, L.; PALAMIDESSI, C.; YUNG, M. (Eds.). **Automata, Languages and Programming**. Berlin, Heidelberg: Springer, 2005. p. 1043-1050.

KARP, R. M. Reducibility among combinatorial problems. In: MILLER, R. E.; THATCHER, J. W.; BOHLINGER, J. D. (Eds.). **Complexity of Computer Computations**. Boston: Springer, 1972. p. 85-103.

LAMBRIX, P.; LIU, Q. Debugging the missing is-a structure within taxonomies networked by partial reference alignments. In: **Data & Knowledge Engineering**, v. 86, p. 179-205, 2013.

LAMY, J. B. Owlready: ontology oriented programming in Python with automatic classification and high level construct for biomedical ontologies. In: **Artificial Inteligence in Medicine**, v. 80, p. 11-28.

MEILICKE, C. **Reasoning about ontology mappings in distributed description logics**. Mannheim: University of Mannheim, 2006.

PAPADIMITRIOU, C. H.; STEIGLITZ, K. **Combinatorial Optimization: Algorithms and Complexity**. New York: Dover Publications, Inc, 1998.

SOLIMANDO, A.; JIMÉNEZ-RUIZ, E.; GUERRINI, G. Detecting and correcting conservativity principle violations in ontology-to-ontology mappings. **International Semantic Web Conference**. 2014a.

SOLIMANDO, A.; JIMÉNEZ-RUIZ, E.; GUERRINI, G. A multi-strategy approach for detecting and correcting conservativity principle violations in ontology alignments. **International Workshop on OWL**. 2014b.

SOLIMANDO, A.; JIMÉNEZ-RUIZ, E; GUERRINI, G. Minimizing conservativity violations in ontology alignments: algorithms and evaluation. In: **Knowledge and Information Systems**, v. 51, n. 3, p. 775-819, 2017.

STUDER, R.; BENJAMINS, V. R.; FENSEL, D. Knowledge engineering: principles and methods. Data and knowledge engineering, [s.l.], v. 25, n. 1, p. 161–198, 1998.

ZAMAZAL, O.; SVÁTEK, V. The ten-year OntoFarm and its fertilization within the onto-sphere. In: **Web Semantics: Science, Service and Agents in the World Wide Web**, v.43, p. 46-53, 2017.

ZHANG, S.; MORK, P.; BODENREIDER, O. Lessons learned from aligning two representations of anatomy. **International Workshop on Formal Biomedical Knowledge Representation**. 2004.

ZIMMERMAN, A.; KRÖTZSCH, M.; EUZENAT, J.; HITZLER, P. Formalizing ontology alignment and its operations with category theory. **International Conference on Formal Ontology in Information Systems**. Baltimore, 2006.