A Language for Inconsistency-Tolerant Ontology Mapping

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

by

Kunal Sengupta
B.Tech, DA-IICT, 2006

2015
Wright State University

WRIGHT STATE UNIVERSITY
GRADUATE SCHOOL

August 13, 2015

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY *Kunal Sengupta* ENTITLED *A Language for Inconsistency-Tolerant Ontology Mapping* BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF *Doctor of Philosophy*.

Pascal Hitzler, Ph.D.
Dissertation Director

Arthur Goshtasby, Ph.D.
Director, Computer Science and Engineering Ph.D. Program

Robert E.W. Fyffe, Ph.D.
Vice President for Research and Dean of the Graduate School

Committee on
Final Examination

Pascal Hitzler, Ph.D.

Krzysztof Janowicz, Ph.D.

Krishnaprasad Thirunarayan, Ph.D.

Prabhaker Mateti, Ph.D.

ABSTRACT

Sengupta, Kunal. Ph.D., Department of Computer Science and Engineering, Wright State University, 2015. A Language for Inconsistency-Tolerant Ontology Mapping.

Ontology alignment plays a key role in enabling interoperability among various data sources present in the web. The nature of the world is such, that the same concepts differ in meaning, often so slightly, which makes it difficult to relate these concepts. It is the omni-present heterogeneity that is at the core of the web. The research work presented in this dissertation, is driven by the goal of providing a robust ontology alignment language for the semantic web, as we show that description logics based alignment languages are not suitable for aligning ontologies.

The adoption of the semantic web technologies has been consistently on the rise over the past decade, and it continues to show promise. The core component of the semantic web is the set of knowledge representation languages – mainly the W3C [1] standards Web Ontology Language (OWL), Resource Description Framework (RDF), and Rule Interchange Format (RIF). While these languages have been designed in order to be suitable for the openness and extensibility of the web, they lack certain features which we try to address in this dissertation. One such missing component is the lack of non-monotonic features, in the knowledge representation languages, that enable us to perform common sense reasoning.

For example, OWL supports the open world assumption (OWA), which means that knowledge about everything is assumed to be possibly incomplete at any point of time. However, experience has shown that there are situations that require us to assume that

[1] World Wide Web Consortium - http://w3c.org

certain parts of the knowledge base are complete. Employing the Closed World Assumption (CWA) helps us achieve this. Circumscription is a very well-known approach towards CWA, which provides closed world semantics by employing the idea of minimal models with respect to certain predicates which are closed. We provide the formal semantics of the notion of *Grounded Circumscription*, which is an extension of circumscription with desirable properties like decidability. We also provide a tableaux calculus to reason over knowledge bases under the notion of grounded circumscription.

Another form of common sense logic, is default logic. Default logic provides a way to specify rules that, by default, hold in most cases but not necessarily in all cases. The classic example of such a rule is: If something is a bird then it flies. The power of defaults comes from the ability of the logic to handle exceptions to the default rules. For example, a bird will be assumed to fly by default unless it is an exception, i.e. it belongs to a class of birds that do not fly, like penguins. Interestingly, this property of defaults can be utilized to create mappings between concepts of different ontologies (knowledge bases). We provide a new semantics for the integration of defaults in description logics and show that it improves upon previously known results in literature.

In this study, we give various examples to show the utility and the advantages of using a default logic based ontology alignment language. We provide the semantics and decidability results of a default based mapping language for tractable fragments of description logics (or OWL). Furthermore, we provide a proof of concept system and present a qualitative analysis of the results obtained from the system, and compare it to the results obtained by applying ontology mapping repair techniques.

Contents

List of Figures

List of Tables

Acknowledgment

I was lucky to have the support of so many great people throughout the pursuit of my Ph.D. First of all, I would like to thank my advisor, Dr. Pascal Hitzler, who gave me the opportunity to pursue my Ph.D. under his guidance. I do not have enough words to describe his greatness as an advisor and, more importantly, as a person. It has been a great learning opportunity to work with him.

A special appreciation goes to my dissertation committee members, Dr. Krishnaprasad Thirunarayan, Dr. Prabhaker Mateti, and Dr. Krzysztof Janowicz, for their valuable comments and suggestions to improve this dissertation.

My family has been a great support and source of encouragement throughout this endeavor. I thank my parents, Mr. Biplab Kumar Sengupta and Mrs. Bishakha Sengupta, and my sisters, Sharmila Gupta and Deepali Sengupta, for their love and support, without which, this Ph.D. would not have been possible to achieve.

Thanks a lot to Hema Vijwani who has supported and encouraged me all the way. It would have been difficult to finish this dissertation without her backing.

I would like to thank all of my friends for their support, especially, Pramod Koneru, Shailendrasingh Patil and Mahim Lakhani. I would also like to thank them for helping me when I needed them the most.

The Data Semantics Lab (DaSe Lab) has been a great place to work and I would like to thank all of my colleagues and friends at the DaSe Lab.

This PhD was partially funded by two National Science Foundation (NSF) projects: 1017225 “III: Small: TROn–Tractable Reasoning with Ontologies” and 1440202 “EarthCube Building Blocks: Collaborative Proposal: GeoLink–Leveraging Semantics, and Linked Data for Data Sharing and Discovery in the Geosciences.”

Dedicated to my parents,

Mrs. Bishakha Sengupta and Mr. Biplab Kumar Sengupta.

1 Introduction

The semantic web effort has given rise to a plethora of new domain ontologies across the web. With that arises a significant challenge: aligning the meaning of terms from different data models to enable exchange of data. The nature of the world is such, that the same concepts differ in meaning, often so slightly, that it becomes difficult to relate these concepts. It is the omni-present heterogeneity that is at the core of the web. The process of making connections between various ontologies is commonly known as ontology mapping (ontology alignment) in the semantic web community.

The main goal of this research work is to produce a basis for a robust ontology mapping language to encode mappings, which are not susceptible to the heterogeneous nature of the world. Such a language would allow us to gauge the similarities as well as the differences in the meaning of the same conceptual entities defined in various data sources. The major problem that is, more often than not, encountered as a result of mapping terms from different ontologies, using the standard description logic (DL) based languages, is the problem of logical inconsistency of merged knowledge bases as well as incoherent concepts.[1] The usual approach of dealing with issues that result from mappings are commonly known as ontology mapping repair. Typically, a repair process identifies the minimal set of mappings that cause inconsistencies (or incoherence) and remove the mappings as part of the repair process. We want to take an alternative approach, whereby, we propose a mapping language which makes use of non-standard semantics to avoid such inconsistencies but at

[1] See section 2.3 for definitions

the same time provide more intuitive mappings.

The language should provide new constructs to relate concepts/roles, in a manner that the semantics of these constructs allow the occurrence of exceptions to the mapping relation. This would allow us to say that most people of type *MarriedMale*, in one ontology, are same as that of *MarriedMale* of another, however, there may be exceptions. There are some non-monotonic logics that fit this requirement right away. Especially, default logic which allows us to write inference rules that hold in most cases and at the same time allows for exceptions.

Therefore, as a part of this research, we examine the utility of some non-monotonic extensions of DLs as an ontology mapping language, and in the process, fix some of the issues that occur when we try to extend DLs with these non-monotonic formalisms. In the following, we discuss the problem statement and the approach taken for this research work.

1.1 Problem Statement

As mentioned above, the motive of this research work is to facilitate interoperability between various heterogeneous data sources, by providing an ontology mapping language that is resilient and less vulnerable to the slight differences in the meaning of the terms used in different ontologies. The hypothesis of this dissertation is as follows:

1. Making use of non-monotonic constructs, inspired by default logic can provide an ontology mapping language which improves upon the existing mapping languages based on OWL constructs (and thereby DLs), as they are too strict to be used for reasoning.

2. Consider a query answering set up with multiple low-level ontologies and one over-arching ontology such that only uni-directional (from lower-level ontology to the over-arching ontology) mappings are allowed in the form of default rules. Then,

computing the completion of the over-arching ontology under the semantics of defaults is a decidable problem. Furthermore, we do not impose any restrictions in the application of the default rules to named or implicit individuals.

1.2 Approach

Our approach towards achieving the goal of robust ontology alignment language has mainly two parts. We discuss them in the following.

1.2.1 Exploring Non-monotonic Extensions of Description Logics

The first part of the research work is identifying a suitable non-monotonic language, which helps us solve the problems that one faces when mapping ontologies using the standard monotonic languages. Of course, default logic provides a natural way of modeling mapping relations between concepts, as well as roles of two or more ontologies. Please refer to section 2.4.1 for a preliminary understanding of defaults. Briefly, defaults are a kind of inference rules which state things that are most likely true. E.g., most people have their heart on the left-hand side is a type of default rule. If we know that x is a Human then we can assume that x's heart is on the left-hand side of their body unless it is explicitly stated to be otherwise.

This gives us an intuition that default rules could be made use of to encode mappings. E.g., assume there are two ontologies representing the domain of food, and it is possible that the two ontologies have a concept or a role which look exactly the same prima facie, but upon digging harder, subtle differences in their actual semantics may be discovered. For instance, the definition of concepts named *Vegetarian* and *Non-Vegetarian* defined in both the ontologies, may slightly vary in the meaning, such that in most of the properties they agree, but for some properties, they may differ. In one ontology *Vegetarian* includes *EggEaters* but not in the other ontology, furthermore, the other ontology considers *EggEaters* disjoint

with *Vegetarian*. Clearly, mapping the above concepts, using description logic constructs would render the merged ontology inconsistent. However, specifying the mappings using default rules would allow for exceptions, such that instances of *EggEaters* would be exceptions to the mapping: most instances of *a:Vegetarian* also belong to *b:Vegetarian*, where *a*, *b* are the two food ontologies. It seems very reasonable to model such mappings using defaults.

The qualities that we need from our new language are: (1) Ability to model mappings as default statements, (2) easy integrations with description logics, and (3) decidability of reasoning tasks. The first quality is obvious from the previous discussion. The second quality is needed because we are interested in mapping ontologies in description logics (or OWL). Decidability is also important without which we cannot achieve anything practical.

In this work, we have explored the integrations of description logics with two non-monotonic approaches: default logic and circumscription, to achieve the above mentioned qualities in a mapping language. While defaults is a natural fit to our needs, we explored circumscription as well, the reason being that using circumscription we can simulate defaults (see section 2.4.3) and it also provides a simpler semantics.

Circumscription is an approach to closed world semantics where we can close certain parts (concepts/roles), of the knowledge base such that the extensions of the closed predicates may contain only those things that are necessarily required. The models of a circumscribed knowledge base are minimal, with respect to the extensions of the closed predicates. We provide formal definitions in section 2.4.3. We also present a new semantics called *grounded circumscription* for description logics, which overcomes the previously known problem of undecidability, when roles are closed. In grounded circumscription, along with the condition of minimality for the closed predicates, we restrict their extensions to contain named individuals (or pairs).

Similarly, for defaults it was shown in [6] that integration of defaults and description logics lead to undecidability. We provide an extension of defaults called *free defaults*, that

improves upon the undecidability result and provides a more intuitive semantics towards defaults and description logics.

Both the approaches impose certain restrictions to achieve decidability of the integrated logics. In this work, we explore these two approaches and identify some of the decidable fragments of these languages, which have more desirable features than in the state of the art results.

1.2.2 Mapping Language for Tractable Description Logics

In the first part of the research work, we provide decidability results for description logics with defaults and circumscription. In the second part of our approach, we define the semantics of a default based mapping language for tractable fragments of DL. [2] We assume a restricted, yet very practical, query answering system where there are several heterogeneous data models in the form of (lower-level) ontologies and one overarching ontology. Queries can be asked in terms of the vocabulary of the overarching ontology.

To achieve this, we propose a new mapping language that we develop in this work, using which, one can generate one-way mappings from the lower-level ontologies to the overarching ontology. These mappings are non-monotonic in nature and are mainly based on an integration of defaults with a tractable fragment of DL $\mathcal{ER}_{\bot,\mathcal{O}}$ that we present in section 5.2. $\mathcal{ER}_{\bot,\mathcal{O}}$ is a very simple language consisting of just existentials, conjunction, concept disjointness and assertions. We show that in the query answering scenario considered here, reasoning with default rules integrated with DLs is a decidable problem, without any restrictions placed on the application of defaults.

[2] Tractable DLs are those for which the time complexity of performing reasoning tasks is polynomial with respect to the input size.

1.3 Structure

This document is structured as follows: in Chapter 2, we provide the basic foundations as preliminaries for the rest of the chapters, which includes a brief background on semantic web technologies, syntax and semantics for description logics, as well as basic foundations for circumscription and defaults. Chapter 3 contains the details of grounded circumscription approach including semantics, decidability result as well as a tableau procedure. In Chapter 4, we provide the details of free defaults, its semantics and the decidability results. Chapter 5, we present a default based ontology alignment language for a specific query answering scenario. In Chapter 6, we provide implementation details and the results of a simple evaluation. A discussion of the related work in detail is covered in Chapter 7. Finally, we conclude in Chapter 8.

2 Background and Preliminaries

In this chapter, we familiarize the readers with the background regarding the general area of knowledge representation and reasoning for the Semantic Web. For this, we need to define the notions, as well as the notations, that will be used in all the succeeding chapters.

2.1 Semantic Web and Ontologies

Semantic web is a term that is used to collectively describe a stack of technologies that support the, so called, *Web of data*. The phrase ”web of data” generally refers to sharing data on the web, such that its meaning can be readily understood by a software agent. Before the adoption of semantic web technologies most of the documents present on the web mainly consisted of unstructured data that could be understood only by humans, e.g., HTML documents. Due to the unstructured nature of these documents, the only good way to make queries over the web is through keywords. The idea of the semantic web was proposed to overcome these difficulties. How should the content producers annotate their data, such that it is readily available to be consumed by a software? For this, a set of standards need to be established that could be used to share data, not only with plain text but also with attached meaning. The World Wide Web Consortium (W3C)[1] is an organization that provides these standards, which enables the content producers to share their data in machine-processable format. This brings us to the question: what exactly is needed to

[1] http://www.w3c.org

establish the standards for the semantic web? In figure 2.1,[2] we show the semantic web technology stack, which represents the set of technologies that are required in order to have a standardized way of sharing data. We briefly describe each layer of this stack:

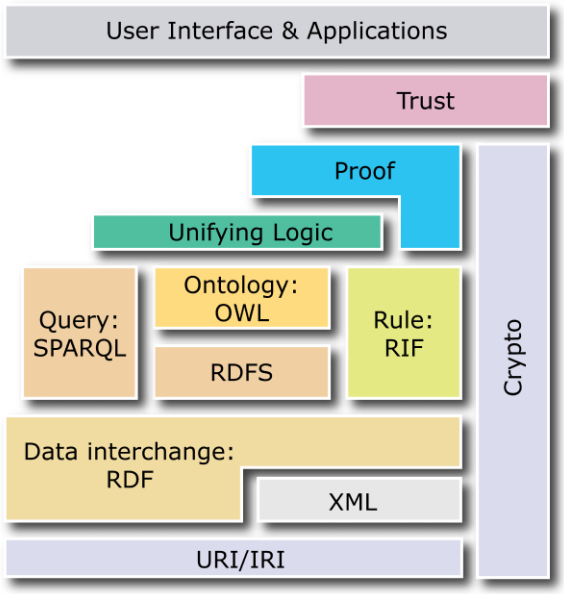

Figure 2.1: The semantic web layer cake diagram.

(I) **URI/IRI**: The most fundamental component of the semantic web technologies is Unique Resource Identifier (URI)/International Resource Identifier (IRI). URI/IRIs are used as identifiers for entities on the web. E.g., to represent my identity on the web, we could use http://example.com/kunal_sengupta as an IRI. It may appear that URI/IRIs are similar to URLs, they indeed look alike, but the URI/IRIs don't need to be resolvable. Given the nature of the web it is possible and most likely, that a single entity has many different URI/IRIs. There are many ways to resolve this, if at all needed, but the discussion on that is out of scope of this document.

(II) **RDF, XML (Data Interchange)**: The second layer on the stack mainly comprises of the serialization formats for data interchange for the semantic web. XML is a

[2]Source: http://www.w3.org/2007/03/layerCake.png

standard data exchange format that can be used to transfer data to, and from, different systems in a platform-independent manner. RDF is a data representation format in graph structure. A statement in RDF is represented as a triple, <subject, predicate, object>. RDF provides its own vocabulary to define classes and their instances. We recommend the reader to visit [39, 68] for a detailed understanding of RDF.

(III) **OWL, RDFS, RIF, and SPARQL**: The third layer of the stack represents the set of W3C standards for the knowledge representation languages, as well as a query language for the semantic web. Knowledge representation languages provide the means to model domain specific knowledge. There are three knowledge representation standards: RDF Schema (RDFS), Web Ontology Language (OWL), and Rule Interchange Format (RIF). The main defining features of a knowledge representation language include the expressiveness with which things can be modeled, the reasoning services it offers, and the complexity of performing the reasoning tasks. RDFS is a lightweight modeling language that provides simple constructs, like classes, objects and relationship between objects using properties. Using the vocabulary of RDFS, we can model domain and range of properties. RDFS is somewhat limited in the area of reasoning, i.e., the amount of implicit knowledge that can be derived from the explicit knowledge is very limited. However, OWL is much more expressive than RDFS and provides more intensive reasoning capabilities. OWL is actually a family of languages derived from the well-known subset of First Order Logic (FOL), Description Logics (DLs). Due to the underpinning of formal logics, OWL has the benefit of unambiguous semantics. The OWL family consists of three sublanguages (also called as fragments or profiles). OWL DL, the most expressive of all OWL fragments, is mainly based on the DL $\mathcal{SROIQ}$. The other OWL profiles are OWL EL, OWL RL, and OWL QL which are based on less expressive description logics, thereby exhibit the property of polynomial time complexity for various reasoning tasks. While there are many ways to write OWL syntax, we omit the specifics and

point the readers to [82] for more details. The semantic web standards also include a query language called SPARQL, using which any semantically enriched database can be queried.

The other layers of the semantic web technology stack are very abstract and largely untouched. We are going to skip the discussion on these layers, as they are also not relevant to this dissertation.

Knowledge representation is the backbone of semantic web technologies. Using knowledge representation languages, we can describe various entities and relationships among them. Semantic web has recommended OWL as the ontology (knowledge representation) language. OWL is a very expressive language; using which we can describe complex relationships between complex entities. They are based on a very well-known family of logics called description logics. The advantage of using description logics as a modeling language is that it comes with, so called, reasoning services. Reasoning is a mechanism to derive implicit knowledge from explicit knowledge, and that is one of the major advantages of using formal logics. In the following chapter, we discuss, one of major DL languages $\mathcal{SROIQ}$.

For readers interested in deeper coverage of the topics we discussed above, we refer them to [3] for RDFS, [3, 39] for OWL.

2.2 Description Logics

Description Logics (DLs) is a family of formal logics, which is a decidable fragment of First Order Logic (FOL). The DL $\mathcal{SROIQ}$ provides the basis for the Web ontology language, which formally defines the semantics of the language. In this section, we first provide the syntax for the description logic $\mathcal{SROIQ}$, followed by its semantics. It should be mentioned that although the major part of the discussion in this dissertation would be using sublanguages of $\mathcal{SROIQ}$, it will be helpful to review $\mathcal{SROIQ}$ in order to understand all the different constructs in the sublanguages. For a deeper coverage of description logics

please refer to [5].

2.2.1 Building Blocks (Syntax)

The syntax of a language consists of the legal constructs that can be used to define terms and relations between them. The fundamental building blocks of a DL language are concepts, roles, and individuals.[3] We define N_C, N_R, and N_I as the mutually disjoint sets of concepts, roles and individuals, respectively. For readers familiar with FOL, concepts are the same as unary predicates, roles are the same as binary predicates, and individuals are the same as constants in FOL. The grammar of the language $\mathcal{SROIQ}$ is defined as follows:

$C = \top \mid \bot \mid A \mid \{a\} \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.D \mid \forall R.D \mid \geq nR.D \mid \leq nR.D \mid \exists R.\mathit{Self}$

Where $A \in N_C$ is called an atomic concept or a classname, $a \in N_I$ is a named individual and $n \in \mathbb{N} \setminus \{0\}$. The symbols $\top$ (top) and $\bot$ (bottom) denote the class of everything and the class of nothing, respectively. Using the above grammar we can define what is usually known as complex class expressions.

Example 1. *The following statements give an intuition of the kind of class expressions we can form using $\mathcal{SROIQ}$ constructs.*

$$\mathit{Human} \sqsubseteq \mathit{Male} \sqcup \mathit{Female} \tag{2.1}$$

$$\mathit{Male} \sqcap \mathit{Female} \sqsubseteq \bot \tag{2.2}$$

$$\mathit{Father} \sqsubseteq \mathit{Male} \sqcap \exists \mathit{hasChild}.\mathit{Human} \tag{2.3}$$

A $\mathcal{SROIQ}$ knowledge base (or ontology) is constructed using a set of axioms. Axioms in a knowledge base can be divided into three sets TBox, RBox, and ABox. A TBox consists of axioms of the form $C \sqsubseteq D$, which we call general concept inclusion (GCI)

[3] Concepts and roles are also referred to as classes and properties respectively.

axioms, where C, D are complex class expressions. The axioms in Example 1 are some examples of GCIs. A RBox consists of role inclusions (RI) of the form: $R \sqsubseteq S$, $R_1 \circ R_2 \sqsubseteq R$, and $\mathsf{Disjoint}(R, S)$. Note that $C \equiv D$ is short for $C \sqsubseteq D$ and $D \sqsubseteq C$ and similarly, $R \equiv S$ is short for $R \sqsubseteq S$ and $S \sqsubseteq R$. An ABox comprises of assertions of the form $C(a), R(a, b)$, $a = b$, and $a \neq b$, where $a, b \in N_I$ are named individuals. In addition to this, $\mathcal{SROIQ}$ also provides inverse role (R^-) constructs, as well as a universal role $\mathcal{U}$ analogous to the concept of $\top$. To retain decidability, certain role regularity constraints are imposed on the role chains; we omit the discussion of role regularity as it is not relevant to this dissertation and point the interested readers to [42].

Example 2.

$$\mathit{hasFather} \circ \mathit{hasBrother} \sqsubseteq \mathit{hasUncle} \tag{2.4}$$

$$\mathit{hasWife} \sqsubseteq \mathit{hasHusband}^- \tag{2.5}$$

$$\mathit{Father}(\mathit{james}) \tag{2.6}$$

$$\mathit{hasBrother}(\mathit{james,peter}) \tag{2.7}$$

Finally, we provide a formal definition for a knowledge base in $\mathcal{SROIQ}$.

Definition 1. *A $\mathcal{SROIQ}$ knowledge base $\mathcal{KB}$ is a finite collection of TBox, RBox and ABox axioms.*

2.2.2 Semantics

The semantics of a language defines the formal meaning of the constructs. Description logic follows model theoretic semantics, in this section we define the notions required to provide a formal understanding of the various axioms in the language of $\mathcal{SROIQ}$.

Table 2.1: Semantics of the language $\mathcal{SROIQ}$

Term	Syntax	Semantics
Individual	a	$a^{\mathcal{I}}$
Top	$\top$	$\Delta^{\mathcal{I}}$
Bottom	$\bot$	$\emptyset$
Nominal	$\{a\}$	$\{a^{\mathcal{I}}\}$
Negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Existential Restriction	$\exists R.D$	$\{x \mid \text{there exists some } y \text{ with } (x,y) \in R^{\mathcal{I}} \text{ and } y \in D^{\mathcal{I}}\}$
Universal Restriction	$\forall R.D$	$\{x \mid \text{for all } y (x,y) \in R^{\mathcal{I}} \rightarrow y \in D^{\mathcal{I}}\}$
At least Restriction	$\geq nR.D$	$\{x \mid \#\{y \text{ with } (x,y) \in R^{\mathcal{I}} \text{ and } y \in D^{\mathcal{I}}\} \geq n\}$
At most Restriction	$\leq nR.D$	$\{x \mid \#\{y \text{ with } (x,y) \in R^{\mathcal{I}} \text{ and } y \in D^{\mathcal{I}}\} \leq n\}$
Role Chain	$R_1 \circ R_2$	$R_1^{\mathcal{I}} \circ R_2^{\mathcal{I}}$
Inverse Role	R^-	$\{(x,y) \mid (y,x) \in R^{\mathcal{I}}\}$
Universal Role	$\mathcal{U}$	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Definition 2. *(Interpretation) An interpretation $\mathcal{I}$ of a $\mathcal{SROIQ}$ knowledge base KB is a pair $(\Delta, .^{\mathcal{I}})$. Where, Δ is non-empty set of interpretation and $.^{\mathcal{I}}$ is a function which maps:*

– *Every named individual a in KB to $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.*

– *Every concept $C \in$ KB to $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$.*

– *Every role $R \in$ KB to $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$*

The syntax and semantics of the DL $\mathcal{SROIQ}$ is summarized in Table 2.1. If an axiom α is satisfied by an interpretation $\mathcal{I}$, we denote that with $\mathcal{I} \models \alpha$. An interpretation $\mathcal{I}$ satisfies a GCI of the form $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds, a RI $R \sqsubseteq S$ if $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ holds. ABox axioms of the form $C(a)$ and $R(a,b)$ are satisfied by $\mathcal{I}$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ hold, respectively. Furthermore, $\mathcal{I}$ satisfies axioms of the form $a = b$, $a \neq b$, and $\mathsf{Disjoint}(R,S)$, if $a^{\mathcal{I}} = b^{\mathcal{I}}$, $a^{\mathcal{I}} \neq b^{\mathcal{I}}$, and $R^{\mathcal{I}} \cap S^{\mathcal{I}} = \emptyset$, respectively

Definition 3. *(Model) Given a $\mathcal{SROIQ}$ knowledge base KB, let $\mathcal{I}$ be an interpretation of KB. Then $\mathcal{I}$ is said to be a model of KB ($\mathcal{I} \models$ KB) if $\mathcal{I}$ satisfies all axioms $\alpha \in$ KB. In other words $\mathcal{I} \models$ KB iff for all $\alpha \in$ KB, $\mathcal{I} \models \alpha$ holds.*

$$\Delta^{\mathcal{I}} = \{a, b, c\} \qquad hasFather^{\mathcal{I}} = \{(c, a)\}$$
$$james^{\mathcal{I}} = a \qquad hasChild^{\mathcal{I}} = \{(a, c)\}$$
$$peter^{\mathcal{I}} = b \qquad hasBrother^{\mathcal{I}} = \{(a, b)\}$$
$$Human^{\mathcal{I}} = \{a, b, c\} \qquad hasUncle^{\mathcal{I}} = \{(c, b)\}$$
$$Male^{\mathcal{I}} = \{a, b\} \qquad hasWife^{\mathcal{I}} = \emptyset$$
$$Female^{\mathcal{I}} = \{c\} \qquad hasHusband^{\mathcal{I}} = (hasHusband^{-})^{\mathcal{I}} = \emptyset$$
$$Father^{\mathcal{I}} = \{a\}$$

Figure 2.2: An interpretation $\mathcal{I}$ of a knowledge base KB

This brings us to the notions of satisfiability and logical consequence of a $\mathcal{SROIQ}$ knowledge base.

Definition 4. *(Satisfiability) Given a $\mathcal{SROIQ}$ knowledge base KB, KB is said to be satisfiable or consistent if there exists a model $\mathcal{I}$ of KB. Otherwise, KB is said to be unsatisfiable or inconsistent.*

The notion of a logical consequence is used to define what is logically implied from the knowledge base. This allows us to formally infer implicit knowledge from explicit knowledge.

Definition 5. *(Logical Consequence) Given a $\mathcal{SROIQ}$ knowledge base KB, an axiom α is said to be a logical consequence of KB ($KB \models \alpha$) iff $\mathcal{I} \models \alpha$ holds for all models $\mathcal{I}$ of KB. We also say that KB entails α.*

Example 3. *Consider a $\mathcal{SROIQ}$ knowledge base KB consisting of all the axioms from Examples 1, 2. $\mathcal{I}$ is an interpretation of KB defined in Figure 2.2. It can be seen that the interpretation $\mathcal{I}$ is indeed a model of the knowledge base KB as it satisfies all the axioms in KB.*

Note that in this example, we have an individual which is not explicitly mentioned in the knowledge base. We refer to such individuals as unknown/anonymous/unnamed individuals. These individuals are produced due to existential restrictions. As in the case of this example we had the existential *Father* $\sqsubseteq$ *Male* $\sqcap \exists$*hasChild.Human*, since *james* is a *Father* it entails that there is an unnamed individual *c* who is a child of *james*.

One of the major advantages of using formal semantics is logical reasoning. Logical reasoning is a mechanism to infer implicit knowledge from explicit knowledge using the semantics of the underlying language. Now, we discuss the kind of inferences that one can derive from a $\mathcal{SROIQ}$ knowledge base as follows:

– Consistency checking: Given a DL knowledge base KB, is it consistent?

– Subsumption checking: Given a DL knowledge base KB, does $KB \models C \sqsubseteq D$?

– Instance checking: Given a DL knowledge base KB, does $KB \models C(a)$?

– Instance retrieval: Given a DL knowledge base KB, find all individuals a such that $KB \models C(a)$.

– Classification: Computing the subsumption hierarchy for a given DL knowledge base KB.

Finally, in this section we provide a theorem which forms the basis for how the reasoners work to produce logical consequences. As it can be observed, there are many possible models of a knowledge base, and it would be virtually impossible to check all the models for logical consequences. Decision procedures for DLs are mainly implemented as a Tableaux procedure. We discuss a Tableaux algorithm in this dissertation in the later chapters. A Tableaux procedure constructs a set of canonical models to check for the logical consequences.

Theorem 1. *Let KB be a $\mathcal{SROIQ}$ knowledge base, then $KB \models \alpha$ iff $KB \cup \{\neg\alpha\}$ is unsatisfiable. Where α is a $\mathcal{SROIQ}$ axiom.*

It is worth mentioning here that description logics, as well as OWL, have the property on monotonicity. The property of monotonicity can be formally defined as: Let KB and KB' be two knowledge bases, such that $KB \subseteq KB'$ then $\{\alpha \mid KB \models \alpha\} \subseteq \{\alpha \mid KB' \models \alpha\}$. In other words adding new axioms cannot invalidate previously derived logical consequences.

2.3 Ontology Alignment

In the previous section, we saw how knowledge can be represented using description logics; in this section we will see a way to integrate different ontologies. Ontology alignment (or mapping) [48, 88] is the notion of creating correspondences between terms from different ontologies (knowledge bases). This is a very active area of research where many researchers are trying to come up with ways to (semi-)automatically match ontologies. The task of ontology alignment is to identify relations between semantically similar or same terms defined in different ontologies. This is important to the data sharing community on the web, as it will improve overall interoperability among different data sources.

Definition 6. *Let $\mathcal{O}_1, \mathcal{O}_2$ be two DL ontologies, a mapping is defined as a quadruplet represented as $< P_{\mathcal{O}_1}, P_{\mathcal{O}_2}, \mathcal{R}, m >$. Where, $P_{\mathcal{O}_1}, P_{\mathcal{O}_2}$ are both classes or roles from $\mathcal{O}_1, \mathcal{O}_2$, respectively, $\mathcal{R} = \{\sqsubseteq, \sqsupseteq, \equiv\}$, and $0 \leq m \leq 1$ is a confidence value. An ontology mapping is then defined as the triplet $< \mathcal{O}_1, \mathcal{O}_2, \delta >$, where δ is a set of all the mappings.*

It can be seen from our definition that the concepts and roles from different ontologies can be mapped using monotonic constructs available in DLs. In later chapters, we show that using DL constructs to map ontologies can lead to problems. The reason being, identifying ontology alignments is a complex procedure and often depends on the syntactic structure of the terms, but due the heterogeneity present in the web, it may happen that the automatically mapped terms are semantically slightly different. We elaborate on this point in later chapters.

Now, we identify two of the major issues that occur when using monotonic DL constructs to map ontologies.

Definition 7. *(Inconsistent mapping) Given two DL ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, and $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, the ontology mapping between $\mathcal{O}_1$ and $\mathcal{O}_2$. A mapping is called* inconsistent *if $\mathcal{O}_1 \cup \mathcal{O}_2$ is consistent and $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \overline{\delta}$ is inconsistent. $\overline{\delta}$ is a set of axioms such that $\mathcal{P}_{\mathcal{O}_1} R\, \mathcal{P}_{\mathcal{O}_2} \in \overline{\delta}$ iff $< P_{\mathcal{O}_1}, P_{\mathcal{O}_2}, R, m > \in \delta$.*

Ontology mappings may also lead to mapping incoherence.

Definition 8. *Given two DL ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ and $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, the ontology mapping between $\mathcal{O}_1$ and $\mathcal{O}_2$, a mapping is called* incoherent *if $\mathcal{O}_1 \cup \mathcal{O}_2 \not\models A \sqsubseteq \bot$ and $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \overline{\delta} \models A \sqsubseteq \bot$ hold, where A is a classname in $\mathcal{O}_1 \cup \mathcal{O}_2$ and $\overline{\delta}$ is a set of axioms such that $\mathcal{P}_{\mathcal{O}_1} R\, \mathcal{P}_{\mathcal{O}_2} \in \overline{\delta}$ iff $< P_{\mathcal{O}_1}, P_{\mathcal{O}_2}, R, m > \in \delta$.*

It is indeed possible and in many cases, automatic alignments do lead to inconsistent mappings and incoherent mappings. One approach to resolve these problems is to use the technique of ontology mapping repair.

Definition 9. *(Mapping repair) Let $(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \delta)$ be an incoherent or inconsistent ontology mapping. A mapping $\delta' \subset \delta$ is called a* mapping repair *of $(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \delta)$ if $(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \delta')$ is consistent and not an incoherent mapping.*

It is clear from the definition that some of the mapping axioms need to be removed in order to fix the issues of inconsistency and incoherence. We take an alternative approach where we modify the mapping language such that we do not remove any axioms as such and retain the maximal mapping between the two ontologies. Please refer to Chapters 4 and 5 for more details on our language.

2.4 Non-monotonic Logics

In this section, we are going to familiarize the reader with two different types of non-monotonic logics, which is a branch of formal logic with different properties in their semantics than the monotonic FOL and DLs that give a non-monotonic characteristic to them. There are three major types of non-monotonic logics in literature Default Logic, Circumscription, and Autoepistemic Logic, we discuss the first two in this chapter, as the third one is less relevant to this dissertation.

2.4.1 Default Logic

Default logic was first introduced for first order predicate logic (FOL), by Reiter in the 1980s [80]. The main purpose of the logic was to model logical sentences which could state default behavior of things, such that incomplete knowledge could be completed by making the most probable conjectures. The classical example happens to be of birds– in absence of information to the contrary- if something is a bird then we can assume that it flies. In simple words default logic allows us to model sentences of the form, "Most birds fly". Something that does not fit a default sentence is known as an exception. In this section, we introduce the basic notions of defaults that would be useful in understanding the rest of the chapters. We first define the notion of default rules formally.

Definition 10. *A default rule is an expression of the form* $\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma}$*, where* α, β_i, γ *are first order formulae.* α *is called the* pre-requisite *of the rule,* $\beta_1, \ldots, \beta_n$ *are its* justifications *and* γ *its* consequent*. A default rule is* closed *if all the formulae that occur in the default are closed first order formulae, otherwise the default rule is called* open*. A* default theory *is further defined as a pair* $(\mathcal{D}, \mathcal{W})$*, where* $\mathcal{D}$ *is a set of defaults and* $\mathcal{W}$ *is a set of closed first order formulae. A default theory is* closed *if all the default rules in the set* $\mathcal{D}$

are closed, otherwise it is called an open *default theory.*

The intuitive meaning of a default rule is that if α is true, and if furthermore assuming $\beta_1, \ldots, \beta_n$ to be true does not result in an inconsistency, then γ is entailed.

Example 4. *In this example we show the FOL sentences and default rule for the classical bird example. In statement 2.8, we say every Penguin is also a Bird. In statement 2.9, we say a Penguin does not fly. Followed by a fact saying Tweety is a Bird. While the default rule 2.11 says if something is a Bird and it is safe to assume that it flies then assume it flies.*

$$\forall x(\textit{Penguin}(x) \rightarrow \textit{Bird}(x)) \tag{2.8}$$

$$\forall x(\textit{Penguin}(x) \rightarrow \neg\textit{Fly}(x)) \tag{2.9}$$

$$\textit{Bird}(\textit{Tweety}) \tag{2.10}$$

$$\frac{\textit{Bird}(x) : \textit{Fly}(x)}{\textit{Fly}(x)} \tag{2.11}$$

The formal semantics of a default theory is defined in terms of a notion of an extension. An extension of a default theory is a completion (i.e., closure under entailment) of a possibly incomplete theory. The following formally describes the notion of an extension, directly taken from [80].

Definition 11. . *Let $\Delta = (\mathcal{D}, \mathcal{W})$ be a closed default theory, so that every default of $\mathcal{D}$ has the form*

$$\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma},$$

where $\alpha, \beta_1, \ldots, \beta_n, \gamma$ are all closed formulae of L (a first order language). For any set of closed formulae $S \subseteq L$, let $\Gamma(S)$ be the smallest set satisfying the following three properties:

- $\mathcal{W} \subseteq \Gamma(S)$

- $\Gamma(S)$ *is closed under entailment.*

- *If* $\frac{\alpha : \beta_1, \ldots \beta_n}{\gamma} \in \mathcal{D}$, $\alpha \in \Gamma(S)$, *and* $\neg\beta_1, \ldots, \neg\beta_n \notin \Gamma(S)$, *then* $\gamma \in \Gamma(S)$.

A set of closed formulae $E \subseteq L$ *is an* extension *of* Δ *if* $\Gamma(E) = E$, *i.e. if* E *is a fixed point of the operator* Γ.

The complexity of reasoning with (variants of) default logic is generally very high [34], and the same holds for most other non-monotonic logics, unless severe restrictions are put in place.[4] In this dissertation, we discuss a special kind of default rules, called normal defaults as defined below.

Definition 12. *(Normal Defaults) Let* $\bar{r} = \frac{\alpha : \beta}{\gamma}$ *be a default rule, we say* $\bar{r}$ *is a normal default if* $\beta = \gamma$, *i.e. the justification and the consequent are the same.*

As one can imagine, having many rules in a default theory could lead to multiple extensions as the triggering of one rule could stop some other rule to fire and vice versa. For example, if in a knowledge base we have the following default rules along with the facts $\mathit{Quaker}(\mathit{Nixon}), \mathit{Republican}(\mathit{Nixon})$

$$\frac{\mathit{Quaker} : \mathit{Pacifist}}{\mathit{Pacifist}} \qquad \frac{\mathit{Republican} : \neg\mathit{Pacifist}}{\neg\mathit{Pacifist}}$$

As we can see, that depending on which of the default rules is fired first, we end up with a different extension of the knowledge base:
$\{\mathit{Republican}(\mathit{Nixon}), \mathit{Quaker}(\mathit{Nixon}), \mathit{Pacifist}(\mathit{Nixon})\}$ and
$\{\mathit{Republican}(\mathit{Nixon}), \mathit{Quaker}(\mathit{Nixon}), \neg\mathit{Pacifist}(\mathit{Nixon})\}$. What would be a valid conclusion of this knowledge base $\mathit{Pacifist}(\mathit{Nixon})$ or $\neg\mathit{Pacifist}(\mathit{Nixon})$? There are two ways to

[4] An exception is [55] for tractable description logics, but the practical usefulness of that approach for default modeling still needs to be shown.

have logical conclusions for defaults: (1) Skeptical – α is a logical consequence if, and only if, $\alpha \in E$ for all extensions E of the knowledge base, (2) Credulous – α is a logical consequence if for some extension E of the knowledge base $\alpha \in E$. It should also be noted that, there is a possibility that there is no extension for a default theory, but in the case of *normal defaults*, it has been shown in [80], that there is always an extension for a normal default theory.

2.4.2 Integration with Description Logics

We have now set up the basic information required to understand what default logic is. We briefly discuss the results of integrating defaults with description logics and the relevant problems in doing so. In [6], the authors try to integrate defaults with DLs and call them terminological default theories. It is shown in [6] that, for a certain extension of the DL $\mathcal{ALC}$ the problem of finding extensions of a terminological default theory is undecidable. The problem is alleviated by restricting the application of defaults to named individuals in the knowledge base.

We improve the results of [6] by lifting the restrictions on application of defaults in later chapters. The remaining open question is: what are the maximal decidable combinations of defaults and DLs?

2.4.3 Circumscription

The other main approach for non-monotonic reasoning, which we consider in this dissertation is Circumscription. The original idea of circumscription was given by J. McCarthy for FOL in [69]. The notion of circumscription allows us to close certain predicates (concepts/roles), such that closed world semantics could apply to them. Two different ways of dealing with incomplete information in logics are: *open world assumption* (OWA) and *close world assumption* (CWA). In OWA, we assume that the knowledge is always incom-

plete, and therefore, if a fact is missing from the knowledge base, we don't assume it to be false. Whereas, in CWA, knowledge is assumed to be complete and so the absence of the fact is evidence enough for it to be false. We start by defining the central notions of circumscription and define what a circumscriptive model of a knowledge base is. In the following discussion, we use $\mathcal{L}$ as a place holder for any decidable DL.

Definition 13. *(Circumscriptive Pattern) Given a knowledge base KB in $\mathcal{L}$, we define the triple $CP = (M, V, F)$ as the circumscriptive pattern. Where, M, V, and F are mutually disjoint sets of predicates called minimal, variable, and fixed predicates, respectively.*

Definition 14. *(Circumscribed Knowledge Base) Given a KB in $\mathcal{L}$ and a circumscription pattern $CP = (M, V, F)$ containing predicates from KB only, we call the pair (KB, CP) as a circumscribed knowledge base.*

Note, that in addition to the knowledge base we need to define the CP including the set of minimal predicates, M, containing the predicates to which we want to apply CWA. Fixed predicates are those that need to have the same extensions in the models in order to be compared for minimality. Furthermore, the variable predicates are unrestricted in their extensions. Minimality of models is a key concept in circumscription, and the idea is that in the minimal models the extensions of the minimized (or closed) predicates contain only those individuals that are necessarily required. The notion of interpretation for circumscribed knowledge bases remains the same as mentioned in section 2.2. Now, we provide a preference relation to compare interpretations for minimality.

Definition 15. *(Preference relation $<_{CP}$) Given a circumscribed knowledge base (KB, CP), let $\mathcal{I}$ and $\mathcal{J}$ be to interpretations of this knowledge base. Then we say $\mathcal{I}$ is preferred over $\mathcal{J}$ or $\mathcal{I} <_{CP} \mathcal{J}$ if all of the following hold:*

1. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$

2. $a^{\mathcal{I}} = a^{\mathcal{J}}$, *for all named individuals* a *in KB*
3. $p^{\mathcal{I}} = p^{\mathcal{J}}$, *for all predicates* $p \in F$
4. $p^{\mathcal{I}} \subseteq p^{\mathcal{J}}$, *for all predicates* $p \in M$
5. $p^{\mathcal{I}} \subset p^{\mathcal{J}}$, *for some predicate* $p \in M$

With the definition of a preference relation, we can now define the notion of model for a circumscribed knowledge base.

Definition 16. *Given a circumscribed knowledge base* (KB, CP)*,* $\mathcal{I}$ *is a circumscribed model of* (KB, CP) *if all of the following hold:*

– $\mathcal{I} \models KB$*, as per the underlying DLs definition.*

– $\mathcal{I}$ *is minimal with respect to the preference relation* $<_{CP}$*, i.e. there in no model* $\mathcal{J}$ *of KB such that* $\mathcal{J} <_{CP} \mathcal{I}$ *holds.*

$\mathcal{I} \models_{CP} KB$ *is the symbolic representation of* $\mathcal{I}$ *is a circumscriptive model of KB.*

Circumscription is an attractive, non-monotonic approach because of the simplicity of its semantics and minimalistic modeling needs. However, it is up to the knowledge modeler to identify which predicates need to go into the different sets of the circumscriptive pattern. An interesting aspect of circumscription is that we can also simulate default rules using abnormal predicates as shown in example below.

$$\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}}) \qquad \mathcal{J} = (\Delta^{\mathcal{J}}, .^{\mathcal{J}})$$
$$\Delta^{\mathcal{I}} = \{a, b, c\} \qquad \Delta^{\mathcal{J}} = \{a, b, c\}$$
$$x^{\mathcal{I}} = x, (x = a, b, c) \qquad x^{\mathcal{J}} = x, (x = a, b, c)$$
$$\textit{Bird}^{\mathcal{I}} = \{a, b, c\} \qquad \textit{Bird}^{\mathcal{J}} = \{a, b, c\}$$
$$\textit{Penguin}^{\mathcal{I}} = \{b, c\} \qquad \textit{Penguin}^{\mathcal{J}} = \{c\}$$
$$\textit{Fly}^{\mathcal{I}} = \{a\} \qquad \textit{Fly}^{\mathcal{J}} = \{a, b\}$$
$$\textit{ab}^{\mathcal{I}}_{\textit{Bird}} = \{b, c\} \qquad \textit{ab}^{\mathcal{J}}_{\textit{Bird}} = \{c\}$$

Figure 2.3: Interpretations $\mathcal{I}$ and $\mathcal{J}$ of the knowledge base from Example 5

.

Example 5. *In this example we simulate the birds example from the previous section*

$$\textit{Bird}(a) \tag{2.12}$$

$$\textit{Bird}(b) \tag{2.13}$$

$$\textit{Penguin}(c) \tag{2.14}$$

$$\textit{Penguin} \sqsubseteq \textit{Bird} \tag{2.15}$$

$$\textit{Penguin} \sqcap \textit{Fly} \sqsubseteq \bot \tag{2.16}$$

$$\textit{Bird} \sqcap \neg \textit{ab}_{\textit{Bird}} \sqsubseteq \textit{Fly} \tag{2.17}$$

Furthermore, we assign to the set $M = \{\textit{ab}_{\textit{Bird}}\}$ to minimize the Birds that don't fly.

In the above example, we make use of a new concept, $\textit{ab}_{\textit{Bird}}$, which represents the set of abnormal birds that do not fly. Consider the two interpretations in Figure 2.3. Clearly, both $\mathcal{I}$ and $\mathcal{J}$ are classical models of the knowledge base from Example 5. However, we know for sure that $\mathcal{J}$ is not a circumscriptive model of the knowledge base as $\mathcal{I} <_{CP} J$ because $\textit{ab}^{\mathcal{I}}_{\textit{Bird}} \subset \textit{ab}^{\mathcal{J}}_{\textit{Bird}}$. Although, there are many benefits of using circumscription for CWA, there are certain problems when it comes to the complexity of reasoning. Most non-

monotonic DLs have very high computational complexity for the reasoning tasks compared to their monotonic counterparts. The problem with circumscription goes beyond the complexity. In fact, it has been shown in literature [11] that the inference problems for circumscribed knowledge bases are undecidable when the roles are minimized unless severe restrictions are placed. We provide an alternative semantics to circumscription to avoid the undecidability issue.

3 Grounded Circumscription

In this chapter, we present the semantics, decidability results, and reasoning algorithms for grounded circumscription. The content of this paper is mainly taken from our published paper [84].

3.1 Introduction

The semantics of the Web Ontology Language OWL [38] (which is based on the description logic $\mathcal{SROIQ}$ [39]) adheres to the Open World Assumption (OWA), i.e., statements which are *not* logical consequences of a given knowledge base are not necessarily considered false. The OWA is a reasonable assumption to make in the World Wide Web context (and thus for Semantic Web applications). However, situations naturally arise where it would be preferable to use the Closed World Assumption (CWA), that is, statements which are *not* logical consequences of a given knowledge base are considered false. Such situations include, for example, when data is being retrieved from a database, or when data can be considered *complete* with respect to the application at hand (see, e.g., [35, 79]).

As a consequence, efforts have been made to combine OWA and CWA modeling for the Semantic Web. Knowledge representation languages, which have both OWA and CWA modeling features, are said to adhere to the *Local Closed World Assumption* (LCWA). Most of these combinations are derived from non-monotonic logics, which have been studied in logic programming [40], or on first-order predicate logic [69, 72, 80]. Furthermore, many

of them are of a *hybrid* nature, meaning that they achieve the LCWA by combining, e.g., description logics with (logic programming) rules. Please see [57, Section 4].

On the other hand, there are not that many approaches which provide a seamless (non-hybrid) integration of OWA and CWA, and each of them have their drawbacks. This is despite the fact that the modeling task, from the perspective of the application developer, seems rather simple: Users would want to specify, simply, that individuals in the extension of a predicate should be exactly those which are *necessarily required* to be in it, i.e., extensions should be *minimized*. Thus, what is needed for applications is a simple, intuitive approach to closed world modeling, which caters to the above intuition, and is also sound, complete and computationally feasible.

Among the primary approaches to non-monotonic reasoning, there is one approach which employs the minimization idea in a very straightforward and intuitively simple manner, namely *circumscription* [69]. However, a naive transfer of the circumscription approach to description logics, which was done in [10, 11, 35, 36], has three primary drawbacks.

1. The approach is undecicable for expressive description logics (e.g., for the description logic $\mathcal{SROIQ}$), unless awkward restrictions are put into place. More precisely, it is not possible to have non-empty TBoxes plus minimization of roles, if decidability is to be retained.
2. Extensions of minimized predicates can still contain elements which are not named individuals (or pairs of such, for roles) in the knowledge base, which is not intuitive for modeling (see also [35]).
3. Complexity of the approach is very high.

The undecidability issue (point 1) hinges, in a sense, also on point 2 above. In this chapter, we provide a modified approach to circumscription for description logics, which we call *grounded circumscription* that remedies both points 1 and 2.[1] Our idea is simple,

[1] We are not yet addressing the complexity issue; this will be done in future work.

yet effective: we modify the circumscription approach from [10, 11, 35, 36], by adding the additional requirement that extensions of minimized predicates may only contain named individuals (or pairs of such, for roles). In a sense, this can be understood as porting a desirable feature from (hybrid) MNKF description logics [23, 55, 54, 75], to the circumscription approach. In another (but related) sense, it can also be understood as employing the idea of DL-safety [76], respectively of DL-safe variables [59] or nominal schemas [15, 56, 58].

The structure of this chapter is as follows. In Section 3.2, we introduce the semantics of grounded circumscription. In Section 3.3, we show that the resulting language is decidable. Next, we provide a tableaux calculus in Section 3.4 to reason with grounded circumscription. We conclude with a discussion of further work in Section 3.5.

3.2 Local Closed World Reasoning with Grounded Circumscription

In this section, we describe LCW reasoning with grounded circumscription (GC), and also, revisit the syntax and semantics of the Description Logic $\mathcal{ALC}$ and extend it with GC. Some results shown here also apply to many other description logics besides $\mathcal{ALC}$, and we will point this out in each case.

3.2.1 Grounded Circumscription

We now describe a very simple way for ontology engineers to model local closed world aspects in their ontologies: simply use a description logic (DL) knowledge base (KB) as usual, and augment it with *meta*-information, which states that some predicates (concept names or role names) are *closed*. Semantically, those predicates are considered minimized, i.e., their extensions contain only what is absolutely required, and, furthermore, only con-

tain *known* (or *named*) individuals, i.e., individuals which are explicitly mentioned in the KB. In the case of concept names, the idea of restricting their extensions only to known individuals is similar to the notion of nominal schema [15, 58] (and thus, DL-safe rules [59, 76]) and, also, the notion of DBox [87], while the minimization idea is borrowed from circumscription [69], one of the primary approaches to non-monotonic reasoning.

In the earlier efforts to carry over circumscription to DLs [10, 35, 36], circumscription is realized by the notion of *circumscription pattern*. A circumscription pattern consists of three disjoint sets of predicates (i.e., concept names and role names), which are called *minimized*, *fixed* and *varying* predicates, and a preference relation on interpretations.[2] The preference relation allows us to pick *minimal* models as the *preferred* models, with respect to set inclusion of the extensions of the minimized predicates.

Our formalism here is inspired by one of the approaches described by Makinson in [66], namely restricting the set of valuations to get more logical consequences than what we can get as classical consequences. Intuitively, this approach is a simpler version of the circumscription formalism for DLs, as presented in [11, 36], in the sense that we restrict our attention only to models in which the extension of minimized predicates may only contain known individuals from the KB. Furthermore, the predicates (concept names and role names) in KB are partitioned into two disjoint sets of minimized and non-minimized predicates, i.e., no predicate is considered fixed.[3] The non-minimized predicates would be viewed as varying, in the more general circumscription formalism mentioned above.

The non-monotonic feature of the formalism is given by restricting models of an $\mathcal{L}$-KB, such that the extension of closed predicates may only contain individuals (or pairs of them) which are explicitly occurring in the KB, plus a minimization of the extensions of these predicates. We define a function Ind that maps each $\mathcal{L}$-KB to the set of individual

[2]There is also a notion of *prioritization* which we will not use, mainly because we are not convinced yet that it is a desirable modeling feature for local closed world reasoning for the Semantic Web.

[3]Fixed predicates can be simulated in the original circumscriptive DL approach if negation is available, i.e., for fixed concept names, concept negation is required, while for fixed role names, role negation is required. The latter can be added to expressive DLs without jeopardizing decidability [58, 91].

names it contains, i.e., given an $\mathcal{L}$-KB K, $\mathsf{Ind}(K) = \{b \in \mathsf{N}_I \mid b \text{ occurs in } K\}$. Among all possible models of K that are obtained by the aforementioned restriction to $\mathsf{Ind}(K)$, we then select a model that is minimal w.r.t. concept inclusion or role inclusion, in accordance with the following definition.

Definition 17. *A* GC-$\mathcal{L}$-KB *is a pair (K, M) where K is an $\mathcal{L}$-KB and $M \subseteq \mathsf{N}_C \cup \mathsf{N}_r$. For every concept name and role name $W \in M$, we say that W is* closed with respect to K. *For any two models $\mathcal{I}$ and $\mathcal{J}$ of K, we furthermore say that $\mathcal{I}$ is* smaller than *(or* preferred over*) $\mathcal{J}$* w.r.t. *M, written $\mathcal{I} \prec_M \mathcal{J}$, iff all of the following hold: (i) $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$ and $a^{\mathcal{I}} = a^{\mathcal{J}}$ for every $a \in \mathsf{N}_I$; (ii) $W^{\mathcal{I}} \subseteq W^{\mathcal{J}}$ for every $W \in M$; and (iii) there exists a $W \in M$ such that $W^{\mathcal{I}} \subset W^{\mathcal{J}}$*

The following notion will be helpful.

Definition 18 (grounded model)**.** *Given a GC-$\mathcal{L}$-KB (K, M), a model $\mathcal{I}$ of K is called a* grounded model *w.r.t M if all of the following hold:*

(1) $C^{\mathcal{I}} \subseteq \{b^{\mathcal{I}} \mid b \in \mathsf{Ind}(K)\}$ for each concept $C \in M$; and

(2) $R^{\mathcal{I}} \subseteq \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid a, b \in \mathsf{Ind}(K)\}$ for each role $R \in M$

We now define models and logical consequence of GC-$\mathcal{L}$-KBs as follows:

Definition 19. *Let (K, M) be a GC-$\mathcal{L}$-KB. An interpretation $\mathcal{I}$ is a* GC-model *of (K, M) if it is a grounded model of K w.r.t. M and $\mathcal{I}$ is minimal w.r.t. M, i.e., there is no model $\mathcal{J}$ of K with $\mathcal{J} \prec_M \mathcal{I}$. A statement (GCI, concept assertion, or role assertion) α is a* logical consequence *(a* GC-inference*) of (K, M) if every GC-model of (K, M) satisfies α. Finally, a GC-$\mathcal{L}$-KB is said to be* GC-satisfiable *if it has a GC-model.*

Note, that every GC-model is also a grounded model. Moreover, in comparison with the more general circumscription formalism for DLs as presented in [11, 36], every GC-

model of a KB is also a circumscriptive model,[4] hence every circumscriptive inference is also a valid GC-inference.

To give an example, consider the knowledge base K consisting of the axioms

$$\mathit{hasAuthor}(\mathit{paper1}, \mathit{author1}) \qquad \mathit{hasAuthor}(\mathit{paper1}, \mathit{author2})$$

$$\mathit{hasAuthor}(\mathit{paper2}, \mathit{author3}) \qquad \top \sqsubseteq \forall \mathit{hasAuthor}.\mathit{Author}$$

Consider the ABox statements:

$\neg \mathit{hasAuthor}(\mathit{paper1}, \mathit{author3})$ and

$(\leq 2\ \mathit{hasAuthor}.\ \mathit{Author})(\mathit{paper1})$.[5]

Neither of them is a logical consequence of K under classical DL semantics. However, if we assume that we have complete information on authorship relevant to the application under consideration, then it would be reasonable to *close* parts of the knowledge base in the sense of the LCWA. In the original approach to circumscriptive DLs, we could close the concept name *Author*, but to no avail, but if we close *hasAuthor*, we obtain $(\leq 2\ \mathit{hasAuthor}.\mathit{Author})(\mathit{paper1})$ as a logical consequence. In addition, if we adopt the Unique Name Assumption (UNA), $\neg \mathit{hasAuthor}(\mathit{paper1}\ , \mathit{author3})$ is also a logical consequence of K. Even without UNA, we can still obtain this as a logical consequence if we add the following axioms to K, which essentially forces the UNA:[6]

$A_1(\mathit{author1}); A_2(\mathit{author2}); A_3(\mathit{author3}); A_i \sqcap A_j \sqsubseteq \bot$ for all $i \neq j$. With regard to this example, note that the closure of roles in the original circumscriptive DL approach leads to undecidability [11]. The GC-semantics, in contrast, is decidable even under role closure (see Section 3.3 below) and also yields the desired inferences.

[4] This can be seen, e.g., by a straightforward proof by contradiction.

[5] The semantics is $(\leq n\ R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid |\{y \mid (x,y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}| \leq n\}$; this *qualified number restriction* is not part of $\mathcal{ALC}$, though it makes a very good example without depending on the UNA.

[6] The UNA can be enforced in an $\mathcal{ALC}$ KB by adding ABox statements $A_i(a_i)$, where a_i are all individuals and A_i are new concept names, to the knowledge base, together with all disjointness axioms of the form $A_i \sqcap A_j \sqsubseteq \bot$ for all $i \neq j$.

3.3 Decidability of Grounded Circumscription

As noted earlier, circumscription in many expressive DLs is undecidable [11]. Undecidability even extends to the basic DL $\mathcal{ALC}$ when non-empty TBoxes are considered and roles are allowed as minimized predicates. Such a bleak outlook would greatly discourage useful application of circumscription, despite the fact that there is a clear need of such a formalism to model LCWA.

Our formalism fills this gap by offering a simpler approach to circumscription in DLs that is decidable provided that the underlying DL is also decidable. The decidability result is obtained due to the imposed restriction of minimized predicates to known individuals in the KB as specified in Definition 19. Let $\mathcal{L}$ be any standard DL. We consider the reasoning task of *GC-KB satisfiability*: "given a GC-$\mathcal{L}$-KB (K, M), does (K, M) have a GC-model?" and show in the following that this is decidable.

Assume that $\mathcal{L}$ is any DL, featuring nominals, concept disjunctions, concept products, role hierarchies and role disjunctions. We show that GC-KB satisfiability in $\mathcal{L}$ is decidable if satisfiability in $\mathcal{L}$ is decidable.

Let (K, M) be a GC-$\mathcal{L}$-KB. We assume that $M = M_A \cup M_r$, where $M_A = \{A_1, \ldots, A_n\}$ is the set of minimized concept names and $M_r = \{r_1, \ldots, r_m\}$ is the set of minimized role names. Now define a family of $(n + m)$-tuples as

$$\mathcal{G}_{(K,M)} = \{(X_1, \ldots, X_n, Y_1, \ldots, Y_m) \mid X_i \subseteq \mathsf{Ind}(K), Y_j \subseteq \mathsf{Ind}(K) \times \mathsf{Ind}(K)\}$$

with $1 \leq i \leq n, 1 \leq j \leq m$. Note that there are

$$\left(2^{|\mathsf{Ind}(K)|}\right)^n \cdot \left(2^{|\mathsf{Ind}(K)|^2}\right)^m = 2^{n \cdot |\mathsf{Ind}(K)| + m \cdot |\mathsf{Ind}(K)|^2} \tag{3.1}$$

such tuples, in particular, note that $\mathcal{G}_{(K,M)}$ is a finite set.

Now, given (K, M) and some $G = (X_1, \ldots, X_n, Y_1, \ldots, Y_m) \in \mathcal{G}_{(K,M)}$, let K_G be the $\mathcal{L}$-KB, consisting of all axioms in K together with all of the following axioms, where the A_i and r_j are all the predicates in M—note that we require role disjunction and concept products for this.

$$A_i \equiv \bigsqcup \{a\} \quad \text{for every } a \in X_i \text{ and } i = 1, \ldots, n$$
$$r_j \equiv \bigsqcup (\{a\} \times \{b\}) \quad \text{for every pair } (a, b) \in Y_j \text{ and } j = 1, \ldots, m$$

Then the following result clearly holds.

Lemma 1. *Let (K, M) be a GC-$\mathcal{L}$-KB. If K has a grounded model $\mathcal{I}$ w.r.t. M, then there exists $G \in \mathcal{G}_{(K,M)}$, such that K_G has a (classical) model $\mathcal{J}$ which coincides with $\mathcal{I}$ on all minimized predicates. Likewise, if there exists $G \in \mathcal{G}_{(K,M)}$, such that K_G has a (classical) model $\mathcal{J}$, then K has a grounded model $\mathcal{I}$ which coincides with $\mathcal{J}$ on all minimized predicates.*

Now consider the set

$$\mathcal{G}'_{(K,M)} = \{G \in \mathcal{G}_{(K,M)} \mid K_G \text{ has a (classical) model}\},$$

and note that, this set is finite and computable in finite time since $\mathcal{G}_{(K,M)}$ is finite and $\mathcal{L}$ is decidable. Furthermore, consider $\mathcal{G}'_{(K,M)}$ to be ordered by the pointwise ordering $\prec$ induced by $\subseteq$. Note that, the point-wise ordering of the finite set $\mathcal{G}'_{(K,M)}$ is also computable in finite time.

Lemma 2. *Let (K, M) be a GC-$\mathcal{L}$-KB and let*

$$\mathcal{G}''_{(K,M)} = \{G \in \mathcal{G}'_{(K,M)} \mid G \text{ is minimal in } (\mathcal{G}'_{(K,M)}, \prec)\}.$$

Then (K, M) has a GC-model, if, and only if, $\mathcal{G}''_{(K,M)}$ is non-empty.

Proof. This follows immediately from Lemma 1 together with the following observation: Whenever K has two grounded models $\mathcal{I}$ and $\mathcal{J}$, such that $\mathcal{I}$ is smaller than $\mathcal{J}$, then there exists $G_{\mathcal{I}}, G_{\mathcal{J}} \in \mathcal{G}'_{(K,M)}$ with $G_{\mathcal{I}} \prec G_{\mathcal{J}}$ such that $K_{G_{\mathcal{I}}}$ and $K_{G_{\mathcal{J}}}$ have (classical) models $\mathcal{I}'$ and $\mathcal{J}'$, respectively, which coincide with $\mathcal{I}$, respectively, $\mathcal{J}$, on the minimized predicates. □

Theorem 2. *GC-KB-satisfiability is decidable.*

Proof. This follows from Lemma 2 since the set $\mathcal{G}''_{(K,M)}$, for any given GC-KB (K, M), can be computed in finite time, i.e., it can be decided in finite time whether $\mathcal{G}''_{(K,M)}$ is empty. □

Some remarks on complexity are as follows: assume that the problem of deciding KB satisfiability in $\mathcal{L}$ is in the complexity class C. Observe from equation (3.1) that there are exponentially many possible choices of the $(n+m)$-tuples in $\mathcal{G}_{(K,M)}$ (in the size of the input knowledge base). Computation of $\mathcal{G}'_{(K,M)}$ is thus in $\textsc{Exp}^{\mathrm{C}}$, and subsequent computation of $\mathcal{G}''_{(K,M)}$ is also in $\textsc{Exp}$. We thus obtain the following upper bound.

Proposition 1. *The problem of finding a GC-model (if one exists) of a given GC-$\mathcal{L}$-KB is in $\textsc{Exp}^{C}$, where C is the complexity class of $\mathcal{L}$. Likewise, GC-$\mathcal{L}$-KB satisfiability is in $\textsc{Exp}^{C}$.*

3.4 Algorithms for Grounded Circumscriptive Reasoning

We now present algorithms for reasoning with grounded circumscription. We start with a tableaux algorithm to decide knowledge base GC-satisfiability, and then, discuss how to extend it to other reasoning tasks. For simplicity of presentation, we only consider GC-KB-satisfiability in $\mathcal{ALC}$, but the procedure should be adaptable to other DLs. Inspiration for the algorithm comes from [36, 43].

3.4.1 Decision Procedure for GC-Satisfiability in $\mathcal{ALC}$

The algorithm is a tableaux procedure, as usual, where the expansion rules are defined to be compatible with the semantics of the language, and for easier reference, we call the resulting algorithm *Tableau1*. It starts with an initial graph F_i constructed using the ABox of a given GC-$\mathcal{ALC}$-KB (K, M), such that all known individuals are represented as nodes along with their labels that consist of the concepts that contain them in the ABox. Additionally, links are added for all role assertions using labels that consist of the roles in the ABox assertion axioms. We call this set of nodes and labels the *initial graph*. The creation of the initial graph F_i is described in terms of the following steps called the initialization process:

- create a node a, for each individual a that appears in at least one assertion of the form $C(a)$ in K (we call these nodes *nominal nodes*),
- add C to $\mathcal{L}(a)$, for each assertion of the form $C(a)$ or $R(a, b)$ in K,
- add R to $\mathcal{L}(a, b)$, for each assertion of the form $R(a, b)$ in K,
- initialize a set $T := \{\mathsf{NNF}(\neg C \sqcup D) \mid C \sqsubseteq D \in K\}$.

The algorithm begins with the initial graph F_i along with the sets T and M, and proceeds by non-deterministically applying the rules defined in Table 1, a process which can be understood as creating a candidate model for the knowledge base. The $\longrightarrow_{TBox}$, $\longrightarrow_{\sqcap}$, $\longrightarrow_{\exists}$ and $\longrightarrow_{\forall}$ rules are deterministic rules, whereas the $\longrightarrow_{\sqcup}$, $\longrightarrow_{GC_C}$ and $\longrightarrow_{GC_R}$ rules are non-deterministic rules, as they provide a choice, with each choice possibly leading to a different graph. The algorithm differs from the usual tableaux algorithm for $\mathcal{ALC}$, as it provides extra $\longrightarrow_{GC_C}$ and $\longrightarrow_{GC_R}$ non-deterministic rules, such that the candidate models are in fact *grounded* candidate models as defined in Definition 18. The rules are applied until a clash is detected or until none of the rules is applicable. A graph is said to contain an *inconsistency clash* when one of the node labels contains both C and $\neg C$, or it contains $\bot$, and it is called *inconsistency-clash-free* if it does not contain an inconsistency clash. The

algorithm, by application of the rules upon termination, generates a so-called *completion graph*. A notion of blocking is required to ensure termination, and we define it as follows.

Definition 20 (Blocking)**.** *A non-nominal node x is* blocked

1. *if it has a* blocked *ancestor; or*
2. *if it has a non-nominal ancestor x', such that $\mathcal{L}(x) \subseteq \mathcal{L}(x')$ and the path between x' and x consists only of non-nominal nodes.*

In the second case, we say that x is directly blocked *by the node x'. Note that, any non-nominal successor node of x is also blocked.*

For a GC-$\mathcal{ALC}$-KB (K, M), the tableau expansion rules when applied exhaustively, generate a completion graph which consists of nodes, edges and their labels, each node x of the graph is labeled with a set of (complex or atomic) concepts and each edge (x, y) is labeled with a set of roles.

Lemma 3 (Termination)**.** *Given any GC-$\mathcal{ALC}$-KB (K, M), the tableaux procedure for (K, M) terminates.*

Proof. First note that node labels can only consist of axioms from K in NNF or of subconcepts of axioms from K in NNF. Thus, there is only a finite set of possible node labels, and thus there is a global bound, say $m \in \mathbb{N}$, on the cardinality of node labels.

Now, note the following: (1) The number of times any rule can be applied to a node is finite, since the labels trigger the rules and the size of the labels is bounded by m. (2) The out-degree of each node is bounded by the number of possible elements of node labels of the form $\exists R.C$, since only the $\longrightarrow_{\exists}$ rule generates new nodes. Thus the out-degree is also bounded by m. Further, infinite, non-looping paths cannot occur since there are, at most,

2^m possible different labels, and so the blocking condition from Definition 20 implies that some node along such a path would be blocked, contradicting the assumption that the path would be infinite. (3) While the $\longrightarrow_{GC_C}$ rule and the $\longrightarrow_{GC_R}$ rule delete nodes, they can only change labels of nominal nodes by possibly adding elements to nominal node labels. Since the number of possible elements of node labels is bounded by m, at some stage, application of the $\longrightarrow_{GC_C}$ rule or the $\longrightarrow_{GC_R}$ rule will no longer add anything to nominal node labels, and then no new applications of rules can be enabled by this process.

From (1) and (2) we obtain a global bound on the size of the completion graphs, which can be generated by the algorithm, and from (3) we see that infinite loops due to deletion and recreation of nodes cannot occur. Thus, the algorithm necessarily terminates. □

Before we show that the tableaux calculus is sound and complete, we define a function called read function which will be needed for clarity of the proof and verification of minimality of the models.

Definition 21 (Read Function)**.** *Given an inconsistency-clash-free completion graph F, we define a read function* r *which maps the graph to an interpretation $\mathsf{r}(F) = \mathcal{I}$ in the following manner. The interpretation domain $\Delta^{\mathcal{I}}$ contains all the non-blocked nodes in the completion graph. Further, for each atomic concept A, we set $A^{\mathcal{I}}$ to be the set of all non-blocked nodes x for which $A \in \mathcal{L}(x)$. For each role name R, we set $R^{\mathcal{I}}$ to be the set of pairs (x, y) which satisfy any of the following conditions:*

- *$R \in \mathcal{L}(x, y)$ and y is not blocked; or*
- *x is an immediate R-predecessor of some node z, and y directly blocks z*

The mapping just defined is then lifted to complex concept descriptions as usual.

The second condition is due to the well-known technique of unraveling (see, e.g., [43]): while disregarding blocked nodes, an incoming edge from an immediate R-predecessor, x of the blocked node z, is considered to be replaced by an edge from the predecessor to the node y which directly blocks z. This accounts for the intuition that a path ending in a blocked node stands for an infinite but repetitive path in the model.

Lemma 4 (Soundness)**.** *If the expansion rules are applied to a GC-$\mathcal{ALC}$-KB (K, M), such that they result in an inconsistency-clash-free completion graph F, then K has a grounded model $\mathcal{I} = \mathsf{r}(F)$. Furthermore, the extension $A^{\mathcal{I}}$ of each concept $A \in M$ under $\mathcal{I}$ coincides with the set $\{x \mid x \in A^{\mathsf{r}(F)}\}$, the extension $R^{\mathcal{I}}$ of each role $R \in M$ under $\mathcal{I}$ coincides with the set $\{(x, y) \mid (x, y) \in R^{\mathsf{r}(F)}\}$, and both these sets can be read off directly from the labels of the completion graph.*

Proof. From the inconsistency-clash-free completion graph F, we create an interpretation $\mathcal{I} = \mathsf{r}(F)$ where r is the read function defined in Definition 21. Since the completion graph is free of inconsistency clashes, and the first five expansion rules from Table 5.2 follow the definition of a model from Section 3.2, the resulting interpretation is indeed a model of K.[7] Moreover, the $\longrightarrow_{GC_C}$ and $\longrightarrow_{GC_R}$ rules ensure that the extensions of minimized predicates contain only (pairs of) known individuals. Hence, $\mathsf{r}(F) = \mathcal{I}$ is a grounded model of K w.r.t M, and Definition 21 shows how the desired extensions can be read off from the completion graph. □

Lemma 5 (completeness)**.** *If a GC-$\mathcal{ALC}$-KB (K, M) has a grounded model $\mathcal{I}$, then the expansion rules can be applied to the initial graph F_i of (K, M) in such a way that they lead to an inconsistency-clash-free completion graph F, and such that the following hold.*

[7] This can be proven formally by structural induction on formulas as in [43].

- $\Delta^{\mathsf{r}(F)} \subseteq \Delta^{\mathcal{I}}$
- $a^{\mathsf{r}(F)} = a^{\mathcal{I}}$ *for every nominal node* a
- $W^{\mathsf{r}(F)} \subseteq W^{\mathcal{I}}$ *for every* $W \in M$
- *the extensions, under* $\mathsf{r}(F)$*, of the closed concept and role names can be read off from* F *as in the statement of Lemma 4.*

Proof. Given a grounded model $\mathcal{I}$ for K w.r.t M, we can apply the completion rules to F_i in such a way that they result in an inconsistency-clash-free completion graph F. To do this, we only have to ascertain that, for any nodes x and y in the graph, the conditions $\mathcal{L}(x) \subseteq \{C \mid \pi(x) \in C^{\mathcal{I}}\}$ and $\mathcal{L}(x, y) \subseteq \{R \mid (\pi(x), \pi(y)) \in R^{\mathcal{I}}\}$ are satisfied, where π is mapping from nodes to $\Delta^{\mathcal{I}}$. This construction is very similar to the one in [43, Lemma 6], to which we refer for details of the argument.

The remainder of the statement follows from the fact that the two conditions just given are satisfied, and from the reading-off process specified in Lemma 4. □

We have provided an algorithm that generates a set of completion graphs, and each inconsistency-clash-free completion graph represents a grounded model. In fact, (K, M) is GC-satisfiable if at least one of the completion graphs is inconsistency-clash-free.

Theorem 3. *Let* (K, M) *be a GC-*$\mathcal{ALC}$*-KB. Then* (K, M) *has a grounded model, if, and only, if it is GC-satisfiable.*

Proof. The *if* part of the proof is trivial.

We prove the *only if* part. For any grounded model $\mathcal{I}$, let $|M_{\mathcal{I}}|$ denote the sum of the cardinalities of all extensions of all the minimized predicates in M, and note that, for any two grounded models $\mathcal{I}$ and $\mathcal{J}$ of K w.r.t. M, we have $|M_{\mathcal{J}}| < |M_{\mathcal{I}}|$ whenever $\mathcal{J} \prec_M \mathcal{I}$.

Hence, for any grounded model $\mathcal{I}$ of K w.r.t. M, which is not a GC-model of (K, M), there is a grounded model $\mathcal{J}$ of K w.r.t. M with $\mathcal{J} \prec_M \mathcal{I}$ and $|M_\mathcal{J}| < |M_\mathcal{I}|$. Since $|M_\mathcal{I}| > 0$ for all grounded models $\mathcal{I}$ (and because $\prec_M$ is transitive), we obtain that, given some grounded model $\mathcal{I}$, it is not possible that there is an infinite descending chain of grounded models preferred over $\mathcal{I}$. Consequently, there must be some grounded model $\mathcal{J}$ of K w.r.t. M which is minimal w.r.t. $|M_\mathcal{J}|$ among all models which are preferred over $\mathcal{I}$. This model $\mathcal{J}$ must be a GC-model, since otherwise it would not be minimal.

□

The following is a direct consequence of Lemmas 3, 4, 5, and Theorem 3.

Theorem 4. *The tableaux algorithm Tableau1 presented above is a decision procedure to determine GC-satisfiability of GC-$\mathcal{ALC}$-KBs.*

3.4.2 Inference Problems beyond GC-Satisfiability

Unlike in other description logics, common reasoning tasks, such as concept satisfiability or instance checking, cannot be readily reduced to GC-satisfiability checking.[8] To cover other inference tasks, we need to extend the previously described algorithm. To do this, we first describe a tableaux algorithm *Tableau2* which is a modification of Tableau1, as follows. All computations are done with respect to an input GC-$\mathcal{ALC}$-KB (K, M).

(i) Initialization of Tableau2 is done on the basis of an inconsistency-clash-free completion graph F, as follows. We create a finite set of nodes which is exactly the domain $\Delta^\mathcal{I}$ of a grounded model $\mathcal{I} = \mathsf{r}(F)$. We distinguish between two different kinds of

[8] E.g., say we want to decide whether (K, M) GC-entails $C(a)$. We cannot do this, in general, by using the GC-satisfiability algorithm in the usual way, i.e., by adding $\neg C(a)$ to K with subsequent checking of its GC-satisfiability. This is because, in general, it does not hold that (K, M) does not GC-entail $C(a)$ if $(K \cup \neg C(a), M)$ is GC-satisfiable. This is due to the non-monotonic nature of circumscription.

nodes, the $\mathcal{I}$-*nominal nodes*, which are nodes corresponding to some $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ where a is an individual name, and the remaining nodes which we call *variable nodes*. For initialization, we furthermore add all information from the ABox of K to the graph and create the set T from K, as in the initialization of Tableau1.

(ii) We modify the $\longrightarrow_{\exists}$ rule as follows:

$$\longrightarrow_{\exists} \quad : \quad \textbf{if } \exists R.C \in \mathcal{L}(x), \text{ and } x \text{ has no R-successor } y \text{ with } C \in \mathcal{L}(y)$$
$$\textbf{then} \text{ select an existing node } y \text{ and}$$
$$\text{set } \mathcal{L}(y) := \{C\} \text{ and } \mathcal{L}(x, y) := \{R\}$$

The above change in the $\longrightarrow_{\exists}$-rule enables us to restrict the graph to contain only the nodes it was initialized with, which means new nodes are not created.

(iii) We retain all other completion rules, however, we dispose of blocking.

(iv) We retain the notion of inconsistency clash, and add a new notion of *preference clash* as follows. A graph F' obtained during the graph construction performed by Tableau2 is said to *contain a preference clash with* $\mathcal{I}$ if at least one of the following holds.

- $W^{\mathsf{r}(F')} = W^{\mathcal{I}}$ for each predicate $W \in M$
- $W^{\mathsf{r}(F')} \cap \{a^{\mathcal{I}} \mid a \text{ an individual } \} \not\subseteq W^{\mathcal{I}}$ for some concept name $W \in M$
- $W^{\mathsf{r}(F')} \cap \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid a, b \text{ individuals } \} \not\subseteq W^{\mathcal{I}}$ for some role name $W \in M$

Proposition 2. *Tableau2 always terminates. If it terminates by constructing an inconsistency and preference-clash-free completion graph F', then $\mathsf{r}(F')$ is preferred over $\mathcal{I}$, i.e., it shows that $\mathcal{I}$ is not a GC-model. If no such graph F' is found, then $\mathcal{I}$ has been verified to be a GC-model.*

Proof. Termination is obvious due to the fact that no new nodes are created, i.e., the algorithm will eventually run out of choices for applying completion rules.

Now assume that the algorithm terminates by finding an inconsistency and preference-clash-free completion graph F'. We have to show that $\mathsf{r}(F')$ is preferred over $\mathcal{I}$, i.e., we need to verify the properties listed in Definition 17. $\Delta^{\mathcal{I}} = \Delta^{\mathsf{r}(F')}$ holds because we initiate the algorithm with nodes being elements from $\Delta^{\mathcal{I}}$ and no new nodes are created. In case nodes are lost due to the grounding rules of Tableau2, we can simply extend $\Delta^{\mathsf{r}(F')}$ with some additional elements which are not otherwise of relevance for the model. The condition $a^{\mathcal{I}} = a^{\mathsf{r}(F')}$ for every $a^{\mathcal{I}} \in \Delta^{\mathsf{r}(F')}$ holds because this is how the algorithm is initialized. The remaining two conditions hold due to the absence of a preference clash.

For the last statement of the proposition, note that Tableau2 will non-deterministically find an inconsistency- and preference-clash-free completion graph if such a graph exists. This can be seen in a similar way as done in the proof of Lemma 5. □

We next use Tableau1 and Tableau2 together to create an algorithm which finds GC-models for (K, M) if they exist. We call this algorithm *GC-model finder*. The algorithm is specified as follows, on input (K, M).

1. Initialize and run Tableau1 on (K, M). If no inconsistency-clash-free completion graph is found, then (K, M) has no GC-model and the algorithm terminates. Otherwise let F be the resulting completion graph.
2. Initialize Tableau2 from F and run it. If no inconsistency- and preference-clash-free completion graph is found, then $\mathsf{r}(F)$ is a GC-model of (K, M) and the algorithm terminates with output $\mathsf{r}(F)$. Otherwise, let F' be the resulting completion graph.
3. Set $F = F'$ and go to step 2.

The loop in steps 2 and 3 necessarily terminates, because whenever step 2 finds a completion graph F' as specified, then $\mathsf{r}(F')$ is preferred over $\mathsf{r}(F)$. As argued in the proof of Theorem 3, there are no infinite descending chains of grounded models w.r.t. the *preferred over* relation, so the loop necessarily terminates. The output $\mathsf{r}(F)$ of the GC-

model finder is a GC-model of (K, M), and we call F a *GC-model graph* of (K, M) in this case.

Theorem 5. *On input a GC-$\mathcal{ALC}$-KB (K, M), the GC-model finder creates a GC-model $\mathcal{I}$ of (K, M) if such a model exists. Conversely, for every GC-model $\mathcal{J}$ of (K, M), there exist non-deterministic choices of rule applications in the GC-model finder such that they result in a model $\mathcal{I}$, which coincides with $\mathcal{J}$ on all extensions of minimized predicates.*

Proof. The first statement follows from Propositon 2 together with the explanations already given. The second statement follows due to Lemma 5, since Tableau1 can already create the sought GC-model $\mathcal{I}$. □

We now consider the reasoning tasks usually known as instance checking, concept satisfiability and concept subsumption. We provide a convenient way to utilize the GC-model finder algorithm to solve these problems by use of another notion of clash called entailment clash. The following definition describes the inference tasks and provides the notion of entailment clash for each of them as well:

Definition 22. *For a GC-$\mathcal{ALC}$-KB (K, M).*

- Instance checking*: Given an atomic concept C and an individual a in (K, M), $(K, M) \models_{GC} C(a)$, if, and only if, $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all GC-models $\mathcal{I}$ of (K, M). For instance checking of $C(a)$, a GC-model graph F is said to contain an* entailment clash *if $C \in \mathcal{L}(a)$ in F.*
- Concept satisfiability*: Given an atomic concept C in (K, M), C is GC-satisfiable, if, and only if, $C^{\mathcal{I}} \neq \emptyset$ for some GC-model of (K, M). For checking satisfiability of C, a GC-model graph F is said to contain an* entailment clash *if $C \in \mathcal{L}(x)$ for any node x in F.*

- Concept subsumption*: Given concepts C and D in (K, M), $(K, M) \models_{GC} C \sqsubseteq D$, if, and only if, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all models $\mathcal{I}$ in (K, M). Subsumption can be reduced to concept satisfiability: GC-$\mathcal{ALC}$-KB$(K, M) \models_{GC} C \sqsubseteq D$ if and only if $C \sqcap \neg D$ is not GC-satisfiable.*

We use the following process to solve these inference problems:

To determine if $C(a)$ is entailed by a GC-$\mathcal{ALC}$-KB (K, M), we invoke the GC-model finder until we find a GC-model. If this non-deterministic procedure results in a GC-model graph which does not contain an entailment clash, then $(K, M) \not\models_{GC} C(a)$. If no such GC-model graph can be generated this way, then $(K, M) \models_{GC} C(a)$.

To determine if C is GC-satisfiable, we invoke the GC-model finder until we find a GC-model. If this non-deterministic procedure results in a GC-model graph which contains an entailment clash, then C is satisfiable. If no such GC-model graph can be generated this way, then C is unsatisfiable.

3.5 Conclusion

We have provided a new approach for incorporating the LCWA into description logics. Our approach, grounded circumscription, is a variant of circumscriptive description logics which avoids two major issues of the original approach: Extensions of minimized predicates can only contain named individuals, and we retain decidability even for very expressive description logics while we can allow for the minimization of roles. We have also provided a tableaux algorithm for reasoning with grounded circumscription.

While the contributions in this chapter provide a novel and, in our opinion, very reasonable perspective on LCWA reasoning with description logics, there are obviously also many open questions. A primary theoretical task is to investigate the complexity of our approach. Of more practical relevance would be an implementation of our algorithm with a

substantial evaluation to investigate its efficiency empirically. More work needs to be done in carrying over the concrete algorithm to description logics which are more expressive than $\mathcal{ALC}$.

It remains to investigate the added value and limitations in practice of modeling with grounded circumscription. This will also shed light onto the question of whether fixed predicates and prioritization are required for applications.

Table 3.1: Tableau1 expansion rules for GC-$\mathcal{ALC}$-KBs (K, M). The first five rules are taken directly from the $\mathcal{ALC}$ tableaux algorithm. Input: F_i, T and M.

$\longrightarrow_{TBox}$	:	**if** $C \in T$ and $C \notin \mathcal{L}(x)$ **then** $\mathcal{L}(x) := \mathcal{L}(x) \cup \{C\}$
$\longrightarrow_{\sqcap}$	:	**if** $C_1 \sqcap C_2 \in \mathcal{L}(x)$, x is not blocked, and $\{C_1, C_2\} \not\subseteq \mathcal{L}(x)$ **then** $\mathcal{L}(x) := \mathcal{L}(x) \cup \{C_1, C_2\}$
$\longrightarrow_{\sqcup}$	:	**if** $C_1 \sqcup C_2 \in \mathcal{L}(x)$, x is not blocked, and $\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$ **then** $\mathcal{L}(x) := \mathcal{L}(x) \cup \{C_1\}$ or $\mathcal{L}(x) := \mathcal{L}(x) \cup \{C_2\}$
$\longrightarrow_{\exists}$	:	**if** $\exists R.C \in \mathcal{L}(x)$, x is not blocked, and x has no R-successor y with $C \in \mathcal{L}(y)$ **then** add a new node y with $\mathcal{L}(y) := \{C\}$ and $\mathcal{L}(x, y) := \{R\}$
$\longrightarrow_{\forall}$	:	**if** $\forall R.C \in \mathcal{L}(x)$, x is not blocked, and x has an R-successor y with $C \notin \mathcal{L}(y)$ **then** $\mathcal{L}(y) := \mathcal{L}(y) \cup \{C\}$
$\longrightarrow_{GC_C}$	:	**if** $C \in \mathcal{L}(x), C \in M, x \notin \mathsf{Ind}(K)$ and x is not blocked **then** for some $a \in \mathsf{Ind}(K)$ do 1. $\mathcal{L}(a) := \mathcal{L}(a) \cup \mathcal{L}(x)$, 2. if x has a predecessor y, then $\mathcal{L}(y, a) := \mathcal{L}(y, a) \cup \mathcal{L}(y, x)$, 3. remove x and all incoming edges to x in the completion graph
$\longrightarrow_{GC_R}$	:	**if** $R \in \mathcal{L}(x, y), R \in M$ and y is not blocked. **then** initialize variables $x' := x$ and $y' := y$, and do 1. if $x \notin \mathsf{Ind}(K)$ then for some $a \in \mathsf{Ind}(K), \mathcal{L}(a) := \mathcal{L}(a) \cup \mathcal{L}(x)$, $x' := a$. 2. if $y \notin \mathsf{Ind}(K)$ for some $b \in \mathsf{Ind}(K), \mathcal{L}(b) := \mathcal{L}(b) \cup \mathcal{L}(y)$ and $y' := b$ 3. if $x' = a$ and x has a predecessor z, then $\mathcal{L}(z, a) := \mathcal{L}(z, a) \cup \mathcal{L}(z, x)$. 4. $\mathcal{L}(x', y') := \mathcal{L}(x', y') \cup \{R\}$ 5. if $x' = a$ remove x and all incoming edges to x and if $y' = b$ remove y and all incoming edges to y from the completion graph.

4 Free Defaults

In this chapter, we present the notion of free defaults, an alternative approach to integration of defaults with description logics [83].

4.1 Introduction and Motivation

The wide adoption of linked data principles has led to an enormous corpus of semantically enriched data being shared on the web. Researchers have been building (semi-)automatic matching systems [1, 88] to build links (correspondences) between various conceptual entities, as well as instances in the linked data. These systems are commonly known as ontology matching/alignment systems. The correspondences generated by these systems are represented using some standard knowledge representation language such as the web ontology language (OWL). However, due to the amount of heterogeneity present in the linked data and the web, OWL does not seem to be a completely suitable language for this purpose as we discuss in the following.

One key aspect of the web (or the world) is variety. There are subtle differences in how a conceptual entity and its relation to other entities are perceived depending on the geographical location, culture, political influence, etc. [46]. To give a simple example consider the concept of marriage. In some conservative parts of the world, marriage stands for a relationship between two individuals of opposite genders whereas in other more liberal places the individuals involved may have the same gender. Consider the axioms in Figure

$$a{:}hasWife \sqsubseteq a{:}hasSpouse \quad (4.1)$$
$$symmetric(a{:}hasSpouse) \quad (4.2)$$
$$\exists a{:}hasSpouse.a{:}Female \sqsubseteq a{:}Male \quad (4.3)$$
$$\exists a{:}hasSpouse.a{:}Male \sqsubseteq a{:}Female \quad (4.4)$$
$$a{:}hasWife(\text{a:john}, a{:}mary) \quad (4.5)$$
$$a{:}Male(a{:}john) \quad (4.6)$$
$$a{:}Female(a{:}mary) \quad (4.7)$$
$$a{:}Male \sqcap a{:}Female \sqsubseteq \bot \quad (4.8)$$
$$symmetric(b{:}hasSpouse) \quad (4.9)$$
$$b{:}hasSpouse(b{:}mike, b{:}david) \quad (4.10)$$
$$b{:}Male(b{:}david) \quad (4.11)$$
$$b{:}Male(b{:}mike) \quad (4.12)$$
$$b{:}Female(b{:}anna) \quad (4.13)$$
$$a{:}hasSpouse \equiv b{:}hasSpouse \quad (4.14)$$
$$a{:}Male \equiv b{:}Male \quad (4.15)$$
$$a{:}Female \equiv b{:}Female \quad (4.16)$$

Figure 4.1: Running example with selected axioms.

4.1, let axioms (5.1) to (5.8) represent a part of ontology A (the conservative perspective), and axioms (5.9) to (5.13) represent a part of ontology B (the liberal perspective). It would be safe to assume that an ontology matching system would output the axioms, (5.14) to (5.16), as the correspondences between these two ontologies. This however, leads to a logical inconsistency under OWL semantics when the two ontologies are merged based on the given correspondences: From axioms (5.10), (5.11), (5.14), and (5.3), we derive *a:Female(b:mike)* which together with axioms, (5.12), (5.15), and (5.8), result in an inconsistency as we derive *a:Male(b:mike)* and *a:Female(b:mike)*, while axiom (5.8) states $a{:}Male \sqcap a{:}Female \sqsubseteq \bot$.

This drives the need for an alignment language which could handle such subtle differences in perspectives. We propose an extension of description logics based on defaults to be used as an ontology alignment language. Using the notion of defaults we could restate axiom (5.14) to an axiom which would semantically mean: every pair of individuals in *b:hasSpouse* is also in *a:hasSpouse* (and vice versa), unless it leads to a logical inconsistency. And those pairs which lead to an inconsistency are treated as exceptions to this axiom. In such a setting, the pair *(b:mike, b:david)* would be treated as an exception to the

re-stated axiom and would not cause an inconsistency any more.

A *default* is a kind of inference rule that enables us to model some type of stereotypical knowledge such as "birds usually fly," or "humans usually have their heart on the left side of their chest." Default logic, which formalizes this intuition, was introduced by Ray Reiter [80] in the 80s, and it is one of the main approaches towards non-monotonic reasoning. In fact, it was the starting point for one of the primary approaches to logic programming and non-monotonic reasoning today, namely the stable model semantics [29] and answer set programming [61].

Reiter's approach is so powerful because exceptions to the default rules are implicitly handled by the logic, so that it is not left to the knowledge modeler to know all relevant exceptions and to take care of them explicitly, as is required in OWL-based ontology modeling. In fact, defaults in the general sense of Reiter still appear to be one of the most intuitive ways of formally modeling this type of stereotypical reasoning [35].

Alas, a paper by Baader and Hollunder [6], published almost 20 years ago, seemed to put an early nail into the coffin of default-extended description logics. Therein, the authors show that a certain extension of the description logic $\mathcal{ALC}$[1] becomes undecidable if further extended with Reiter defaults. Since decidability was (and still is) a key design goal for description logics, this result clearly was a showstopper for further development of default-extended description logics. Of course, Baader and Hollunder also provided a quick fix: If we impose that the default rules only apply to known individuals (i.e., those explicitly present in the knowledge base), then decidability can be retained. However, this semantics for defaults is rather counter-intuitive, as it implies that default rules never apply to unknown individuals. In other words: to unknown individuals the defaults do, by default, not apply. Arguably, this is not a very intuitive semantics for defaults.

In this chapter, we show that there is still a path of development for default-extended description logics, and that they may yet attain a useful standing in ontology modeling.

[1] $\mathcal{ALC}$ is a very basic description logic which, among other things, constitutes the core of OWL 2 DL [38, 39].

In fact, we will present a way to extend decidable description logics with defaults which transcends the approach by Baader and Hollunder while retaining decidability: in our approach, default rules do apply to unknown individuals. We refer to the type of default semantics which we introduce as *free defaults*. Indeed, the contributions of this work are (1) A new semantics for defaults (free defaults) in description logics, and thereby OWL, such that the application of defaults are not limited to named individuals in the knowledge base, (2) We show that reasoning under this new semantics is decidable, which improves upon the results shown in [6], (3) Adding default role inclusion axioms also yield a decidable logic, and (4) We introduce the use of free defaults as a basis for a new language for ontology alignment, and show some application scenarios in where defaults could play a major role and show how our approach covers these scenarios.

4.2 Semantics of Free Defaults

In this section, we introduce the semantics of free defaults. We restrict our attention to normal defaults and show that reasoning in this setting is decidable in general when the underlying DL is also decidable. Normal defaults are very intuitive and we observe that there are many applications in practice where normal defaults can be very useful—see section 4.5. We also provide a DL syntax to encode default rules in the knowledge bases. For our purposes, a normal default rule is of the form $\frac{A : B}{B}$, where A and B are class names,[2] i.e., the justification and conclusion of the default rule are the same. For a description logic $\mathcal{L}$, we are going to represent the same rule in the form of an axiom $A \sqsubseteq_d B$, where A and B are $\mathcal{L}$-concepts and $\sqsubseteq_d$ represents (*free*) *default subsumption*. We refer to statements of the form $A \sqsubseteq_d B$ as (*free*) *default rules* or *default axioms*.

Definition 23. *Let $\mathcal{KB}$ be a description logic knowledge base, and let δ be a set of default axioms of the form $C \sqsubseteq_d D$, where C and D are concepts appearing in $\mathcal{KB}$. Then we call*

[2] We will lift this to roles in section 4.4.

the pair (KB, δ) *a* default-knowledge-base.

The semantics of the default subsumption can be informally stated as follows: if $C \sqsubseteq_d D$, then every named individual in C can also be assumed to be in D, unless it results in a logical inconsistency. Also, if $C \sqsubseteq_d D$, then every unnamed individual in C is also in D, i.e., for unnamed individuals $\sqsubseteq_d$ behaves exactly the same as $\sqsubseteq$. Furthermore, we say a named individual a satisfies a default axiom $C \sqsubseteq_d D$ if (1) $a^{\mathcal{I}} \in C^{\mathcal{I}}, D^{\mathcal{I}}$ or (2) $a^{\mathcal{I}} \in (\neg C)^{\mathcal{I}}$. The intuition behind the semantics of free defaults is to maximize the sets of the named individuals that satisfy the default axioms, while maintaining the consistency of the knowledge base.

The following notations will be needed to formalize this intuition for the semantics of free defaults.

Definition 24. *For a default-knowledge-base* (KB, δ)*, we define the following.*

- Ind_{KB} *is the set of all the named individuals occurring in* KB.
- $\mathsf{P}(\mathsf{Ind}_{KB})$ *is the power set of* Ind_{KB}.
- $\mathsf{P}^n(\mathsf{Ind}_{KB})$ *is the set of n-tuples obtained from the Cartesian product:* $\mathsf{P}(\mathsf{Ind}_{KB}) \times \ldots_{\textit{n times}} \times \mathsf{P}(\mathsf{Ind}_{KB})$*, where* n *is the cardinality of* δ.

The notion of interpretation for the default-knowledge-bases (KB, δ) remains the same as that of the underlying DL of the knowledge base KB.[3] Additionally, given an interpretation $\mathcal{I}$, we define $\delta^{\mathcal{I}}$ to be the tuple $(X_1^{\mathcal{I}}, \ldots, X_n^{\mathcal{I}})$, where each $X_i^{\mathcal{I}}$ is the set of interpreted named individuals that satisfy the i^{th} default $C_i \sqsubseteq_d D_i$ in the sense that $X_i^{\mathcal{I}} = (C_i^{\mathcal{I}} \cap D_i^{\mathcal{I}} \cap \Delta_{Ind}^{\mathcal{I}}) \cup ((\neg C_i)^{\mathcal{I}} \cap \Delta_{Ind}^{\mathcal{I}})$ with $\Delta_{Ind}^{\mathcal{I}} = \{a^{\mathcal{I}} \mid a \in \mathsf{Ind}_{KB}\} \subseteq \Delta^{\mathcal{I}}$ being the set of interpreted individuals occurring in the knowledge base. We now need to define a preference relation over the interpretations, such that we can compare them on the basis of the sets of named individuals satisfying each default.

[3] See chapter 2.

Definition 25. *(Preference relation $>_{KB,\delta}$) Given a knowledge base KB and a set of default axioms δ. Let $\mathcal{I}$ and $\mathcal{J}$ be two interpretations of the pair (KB, δ), then we say that $\mathcal{I}$* is preferred over *$\mathcal{J}$ or $\mathcal{I} >_{KB,\delta} \mathcal{J}$ if all of the following hold.*

1. $a^{\mathcal{I}} = a^{\mathcal{J}}$ *for all* $a \in \mathsf{N}_I$
2. $X_i^{\mathcal{I}} \supseteq X_i^{\mathcal{J}}$ *for all* $1 \leq i \leq |\delta|$, *where* $X_i^{\mathcal{I}} \in \delta^{\mathcal{I}}$ *and* $X_i^{\mathcal{J}} \in \delta^{\mathcal{J}}$.
3. $X_i^{\mathcal{I}} \supset X_i^{\mathcal{J}}$ *for some* $1 \leq i \leq |\delta|$, *where* $X_i^{\mathcal{I}} \in \delta^{\mathcal{I}}$ *and* $X_i^{\mathcal{J}} \in \delta^{\mathcal{J}}$.

The concept of a model under the semantics of free defaults would be the one which is maximal, with respect to the above relation.

Definition 26. *(d-model) Given (KB, δ), we call $\mathcal{I}$ a* d-model *of KB with respect to a set of defaults δ, written $\mathcal{I} \models_d (KB, \delta)$, if all of the following hold:*

1. *$\mathcal{I}$ satisfies all axioms in KB.*
2. *$C_i^{\mathcal{I}} \setminus \Delta_{Ind}^{\mathcal{I}} \subseteq D_i^{\mathcal{I}}$, for each $(C_i \sqsubseteq_d D_i) \in \delta$.*
3. *There is no interpretation $\mathcal{J} >_{KB,\delta} \mathcal{I}$ satisfying conditions 1 and 2 above.*

Furthermore, if (KB, δ) has at least one model, then the default knowledge base is said to be d-satisfiable.

The following proposition is obvious from the definition of d-model.

Proposition 3. *If $\mathcal{I}$ is a d-model of the default-knowledge-base (KB, δ), then $\mathcal{I}$ is a classical model of KB.*

For default theories, two types of entailments are usually considered: credulous and skeptical [80]. A logical formula is a credulous entailment if it is true in at least one of the extensions of the default theory. Skeptical entailment requires the logical formula to be true in all the extensions. We follow the skeptical entailment approach, as it fits better to

the description logic semantics.[4]

Definition 27. *(d-entailment) Given a default-knowledge-base* (KB, δ) *and DL axiom* α*,* α *is* d-entailed *by* (KB, δ) *if it holds in all the d-models of* (KB, δ).

4.3 Decidability

In this section, we show that the tasks of checking for d-satisfiability and d-entailment for default-knowledge-bases are decidable in general. Let (KB, δ) be a default-knowledge-base where KB is encoded in a decidable DL $\mathcal{L}$, which supports nominal concept expressions. We show that finding a d-model for (KB, δ) is also decidable. For some $\mathcal{P} = (X_1, \ldots, X_n) \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, let $KB_{\mathcal{P}}$ be the knowledge base that is obtained by adding the following axioms to KB, for each $C_i \sqsubseteq_d D_i \in \delta$:

1. $\overline{X_i} \equiv (C_i \sqcap D_i \sqcap \{a_1, \ldots, a_k\}) \sqcup (\neg C_i \sqcap \{a_1, \ldots, a_k\})$, where $\overline{X_i}$ is the nominal expression $\{x_1, \ldots, x_m\}$ containing exactly all the named individuals in X_i, and $\{a_1, \ldots, a_k\} = \mathsf{Ind}_{KB}$.
2. $C_i \sqcap \neg\{a_1, \ldots, a_k\} \sqsubseteq D_i$, where $\{a_1, \ldots, a_k\} = \mathsf{Ind}_{KB}$.

The first step in the above construction is useful to identify the sets of default-satisfying individuals. The extensions of the $\overline{X_i}$s represent those sets. The second step ensures all the unnamed individuals satisfy the default axioms. Notice that $KB_{\mathcal{P}}$, as constructed using the above rewriting steps, makes it fall under the expressivity of the DL $\mathcal{L}$, and construction of $KB_{\mathcal{P}}$ requires only a finite number of steps since δ is a finite set. Furthermore, we can compute an order on the set $\mathsf{P}^n(\mathsf{Ind}_{KB})$ based on the $\supseteq$-relation, defined as follows: Let $\mathcal{P}_1, \mathcal{P}_2 \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, then $\mathcal{P}_1 \succ \mathcal{P}_2$ iff

1. $X_{1i} \supseteq X_{2i}$ for each $X_{1i} \in \mathcal{P}_1$ and $X_{2i} \in \mathcal{P}_2$ and
2. $X_{1i} \supset X_{2i}$ for some $X_{1i} \in \mathcal{P}_1$ and $X_{2i} \in \mathcal{P}_2$.

[4] Whether credulous entailment is useful in a Semantic Web context is to be determined.

Lemma 6. *Given a default-knowledge-base (KB, δ), if $KB_{\mathcal{P}}$ is classically satisfiable for some $\mathcal{P} \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, then (KB, δ) has a d-model.*

Proof. Let $\mathcal{P}_1 \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, such that $KB_{\mathcal{P}_1}$ has a classical model $\mathcal{I}$. Then there are two possible cases.

In the first case, there is no $\mathcal{P}_x \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, such that $\mathcal{P}_x \succ \mathcal{P}_1$ and $KB_{\mathcal{P}_x}$ has a classical model. In this case, $\mathcal{I}$ satisfies all the conditions of a d-model: (1) $\mathcal{I}$ satisfies all axioms of KB since $KB \subseteq KB_{\mathcal{P}}$. (2) $\mathcal{I}$ satisfies condition 2 of the definition of d-model, this follows from the second step of the construction of $KB_{\mathcal{P}}$. (3) This follows directly from the assumption for this case. So $\mathcal{I}$ is a d-model for (KB, δ) in this case.

The second case is when there is some $\mathcal{P}_x \in \mathsf{P}^n(\mathsf{Ind}_{KB})$ for which there is a classical model $\mathcal{I}$ for $KB_{\mathcal{P}_x}$ and $\mathcal{P}_x \succ P_1$. Again, there are two possibilities as in case of $\mathcal{P}_1$. Either the first case above holds for $\mathcal{P}_x$, or there is some $\mathcal{P}_y \succ \mathcal{P}_x \in \mathsf{P}^n(\mathsf{Ind}_{KB})$ for which the second case holds. In the latter situation, the argument repeats, eventually giving rise to an ascending chain with respect to the order $\succ$ on $\mathsf{P}^n(\mathsf{Ind}_{KB})$. However, since $\mathsf{P}^n(\mathsf{Ind}_{KB})$ is finite, this chain has a maximal element, and thus the first case applies. Therefore, there is a d-model for (KB, δ). □

The following theorem is a direct consequence of Lemma 6 and the finiteness of δ.

Theorem 6. *The task of determining d-satisfiability of default-knowledge-bases is decidable.*

It should be noted that, in the case of Reiter's defaults it is known that for normal default theories an extension always exists, but in the case of free defaults it can be easily seen that there might be some default-knowledge-bases which do not have a d-model. This

is not completely satisfactory, of course. However, at this stage, it is unknown whether a stronger result can be obtained without giving up decidability. Though the notion of d-satisfiability is important for checking that the default-knowledge-base modelled is consistent and can be used for reasoning-based query services, the more interesting problem in the case of default-knowledge-bases is to handle d-entailment inference services. As it can be observed that d-entailment checking is not directly reducible to satisfiability checking of the default-knowledge-base,[5] we define a mechanism of checking d-entailments and show that this is also decidable.

Proposition 4. *Let (KB, δ) be a default-knowledge-base. If $\mathcal{I}$ is a d-model of (KB, δ), then there exists $\mathcal{P} \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, such that $\mathcal{I}$ is a classical model of $KB_{\mathcal{P}}$ and all classical models of $KB_{\mathcal{P}}$ are d-models of (KB, δ).*

Proof. Given $\mathcal{I}$, we construct a $\mathcal{P} \in \mathsf{P}^n(\mathsf{Ind}_{KB})$ as follows: Given a default $C_i \sqsubseteq_d D_i \in \delta$, let X_i be the maximal subset of Ind_{KB}, such that $X_i^{\mathcal{I}} \subseteq (C_i \sqcap D_i)^{\mathcal{I}} \cup (\neg C_i)^{\mathcal{I}}$. Given all these $X_i's$, let $\mathcal{P} = \{X_1, \ldots, X_n\}$.

Clearly, $\mathcal{I}$ is then a classical model of $KB_{\mathcal{P}}$.

Furthermore, since $\mathcal{I}$ is a d-model of (KB, δ), there is no $\mathcal{P}_x \in \mathsf{P}^n(\mathsf{Ind}_{KB})$, such that $KB_{\mathcal{P}_x}$ has a classical model and $P_x \succ \mathcal{P}$. By construction of $KB_{\mathcal{P}}$ all classical models of $KB_{\mathcal{P}}$ satisfy the three d-model conditions for (KB, δ) because (1) $KB \subseteq KB_{\mathcal{P}}$, all axioms of KB are satisfied, (2) the second step of the construction of $KB_{\mathcal{P}}$ ensures the second condition of d-model is satisfied, (3) Since $\mathcal{P}$ satisfies the maximality condition, all classical models of $KB_{\mathcal{P}}$ also satisfy the maximality condition of being a d-model ensured by step one of the construction of $KB_{\mathcal{P}}$. □

[5] This is due to non-monotonicity of the logic.

Consider the two sets:

$$\mathcal{P}_{KB\text{-d}}=\{\mathcal{P} \in \mathsf{P}^n(\mathsf{Ind}_{KB}) \mid KB_{\mathcal{P}} \text{ is classically satisfiable}\}$$

$$\mathcal{P}_{KB\text{-d-model}}=\{\mathcal{P} \in \mathcal{P}_{KB\text{-d}} \mid \mathcal{P} \text{ is maximal w.r.t. } \succ\},$$

and note that they are both computable in finite time. We refer to all the $KB_{\mathcal{P}}$'s, generated from all $\mathcal{P} \in \mathcal{P}_{\text{KB-d-model}}$, as *d-model generating knowledge bases*.

Lemma 7. *A DL axiom α is d-entailed by a default-knowledge-base (KB, δ), iff it is classically entailed by every $KB_{\mathcal{P}}$ obtained from KB, δ, and all $\mathcal{P} \in \mathcal{P}_{KB\text{-d-model}}$.*

Proof. This is a consequence of Proposition 4, since all classical models of each $\{KB_{\mathcal{P}} \mid \mathcal{P} \in \mathcal{P}_{\text{KB-d-model}}\}$ are also the d-models of the knowledge base. □

We assume KB is in a decidable description logic $\mathcal{L}$ that supports nominals and full negation. It is a well-known result that all common inference tasks are reducible to a satisfiability check in DLs that support full negation [14]. Furthermore, $KB_{\mathcal{P}}$ is constructed by adding GCIs involving concept expressions using nominals and conjunctions, so we can safely assume that $KB_{\mathcal{P}}$ also falls under the DL $\mathcal{L}$. Hence, all the d-model generating knowledge bases are in $\mathcal{L}$.

Theorem 7. *(Decidability of d-entailment) Let $\mathcal{L}$ be a decidable DL with full negation and nominal support. Then the tasks of subsumption checking, instance checking, and class satisfiability are decidable for default-knowledge-bases with KB in $\mathcal{L}$.*

Proof. Given a default knowledge base (KB, δ), then by Lemma 7 the inference tasks can be reduced as follows:

- Subsumption: $C \sqsubseteq D$ is d-entailed by (KB, δ) iff $KB_{\mathcal{P}} \cup \{C \sqcap \neg D\}$ is classically unsatisfiable for all $\mathcal{P} \in \mathcal{P}_{\text{KB-d-model}}$.
- Instance checking: $C(a)$ is d-entailed by (KB, δ) iff $KB_{\mathcal{P}} \cup \{\neg C(a)\}$ is classically unsatisfiable for all $\mathcal{P} \in \mathcal{P}_{\text{KB-d-model}}$.
- Class satisfiability: a class C is satisfiable iff $C \sqsubseteq \bot$ is not d-entailed by (KB, δ).

Consider the task of checking $(KB, \delta) \models_d C \sqsubseteq D$. Then $(KB, \delta) \models_d C \sqsubseteq D$, iff $KB_P \cup \{C \sqcap \neg D\}$ is classically unsatisfiable for all $\mathcal{P} \in \mathcal{P}_{\text{KB-d-model}}$. Since $P_{\text{KB-d-model}}$ is finitely computable and checking classical satisfiability is decidable in $\mathcal{L}$, checking the satisfiability for each $KB_{\mathcal{P}}$ is decidable. Hence, checking $(KB, \delta) \models_d C \sqsubseteq D$ is decidable. Similar arguments hold for the other tasks.

□

4.4 Default Role Inclusion Axioms

So far, we have restricted our attention to default concept inclusions. We made this restriction for the purpose of obtaining a clearer presentation of our approach. However, as may be clear by now, we can also carry over our approach to cover default role inclusions, and we discuss this briefly in the following.

We use the notation $R \sqsubseteq_d S$ for free (normal) role defaults. As in the case of default concept inclusion axioms for role defaults, we restrict the exceptions to these defaults to be pairs of named individuals only. The intuitive semantics of $R \sqsubseteq_d S$ is that for every pair (a, b) of named individuals in the knowledge base, if R holds then assume S also holds unless it leads to an inconsistency. For all other pairs of individuals (with at least one unnamed individual), if R holds then S also holds. We extend the definition of default-knowledge-bases and adjust the other definitions in the following:

Definition 28. *Let KB be a knowledge base in a decidable DL and let δ be a set of default axioms of the form $C \sqsubseteq_d D$ or $R \sqsubseteq_d S$, where C, D and R, S are respectively concepts and roles appearing in KB. Then we call (KB, δ) a* default-knowledge-base. *Furthermore:*

- *The definition of $\mathsf{Ind}_{KB}, \mathsf{P}(\mathsf{Ind}_{KB}), \mathsf{P}^n(\mathsf{Ind}_{KB})$ carry over from Definition 24, where n is the number of axioms of the form $C \sqsubseteq_d D$ in δ.*
- *$\mathsf{P}(\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB})$ denotes the power set of $\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB}$.*
- *$\mathsf{P}^m(\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB})$ is the set of m-tuples obtained from the Cartesian product: $\mathsf{P}(\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB}) \times \ldots_{mtimes} \times \mathsf{P}(\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB})$, where m is the number of default role axioms in δ.*

For simplicity of presentation, we assume that δ is arranged, such that all default concept inclusion axioms appear before all default role inclusion axioms. Now, consider the set $\mathcal{D}_{KB} = \mathsf{P}^n(\mathsf{Ind}_{KB}) \times \mathsf{P}^m(\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB})$ which is a set of tuples, where each tuple is of the form $((X_1, \ldots, X_n)\ , (Y_1, \ldots, Y_m))$, such that $(X_1, \ldots, X_n) \in \mathsf{P}^n(\mathsf{Ind}_{KB})$ and $(Y_1, \ldots, Y_m) \in \mathsf{P}^m(\mathsf{Ind}_{KB} \times \mathsf{Ind}_{KB})$. An interpretation for default-knowledge-bases with default role inclusion axioms should now map δ to a tuple as follows: $\delta^{\mathcal{I}} = (\mathcal{X}^{\mathcal{I}}, \mathcal{Y}^{\mathcal{I}}) \in \mathcal{D}_{KB}$, where $\mathcal{X}^{\mathcal{I}} = (X_1^{\mathcal{I}}, \ldots, X_n^{\mathcal{I}})$ and $\mathcal{Y}^{\mathcal{I}} = (Y_1^{\mathcal{I}}, \ldots, Y_m^{\mathcal{I}})$ such that $X_i^{\mathcal{I}} = (C_i^{\mathcal{I}} \cap D_i^{\mathcal{I}} \cap \Delta_{Ind}^{\mathcal{I}}) \cup ((\neg C)^{\mathcal{I}} \cap \Delta_{Ind}^{\mathcal{I}})$ and $Y_j^{\mathcal{I}} = (R_j^{\mathcal{I}} \cap S_j^{\mathcal{I}} \cap (\Delta_{Ind}^{\mathcal{I}} \times \Delta_{Ind}^{\mathcal{I}})) \cup ((\neg R_j)^{\mathcal{I}} \cap \Delta_{Ind}^{\mathcal{I}} \times \Delta_{Ind}^{\mathcal{I}})$, for all $C_i \sqsubseteq_d D_i \in \delta$ and $R_j \sqsubseteq_d S_j \in \delta$. In other words, $X_i^{\mathcal{I}}$ denotes the extension of the named individuals that satisfy the i^{th} default concept axioms, and $Y_j^{\mathcal{I}}$ denotes the extension of pairs of named individuals that satisfy the j^{th} default role axiom.

To ensure the maximal application of the default axioms, we need the preference relation to be adapted to this setting.

Definition 29. *(Preference Relation $>_{KB,\delta}$) Given a knowledge base KB, a set of default axioms δ, let $\mathcal{I}$ and $\mathcal{J}$ be two interpretations of (KB, δ). We say that $\mathcal{I}$ is preferred over $\mathcal{J}$,*

written $\mathcal{I} >_{KB,\delta} \mathcal{J}$*, if,*

1. *conditions 1-4 of Definition 25 hold,*
2. $Y_i^{\mathcal{I}} \supseteq Y_i^{\mathcal{J}}$ *for all* $1 \leq i \leq m$*, where* $Y_i^{\mathcal{I}} \in \mathcal{Y}^{\mathcal{I}}$ *and* $Y_i^{\mathcal{J}} \in \mathcal{Y}^{\mathcal{J}}$*,*
3. $Y_i^{\mathcal{I}} \supset Y_i^{\mathcal{J}}$ *for some* $1 \leq j \leq m$*, where* $Y_i^{\mathcal{I}} \in \mathcal{Y}^{\mathcal{I}}$ *and* $Y_i^{\mathcal{J}} \in \mathcal{Y}^{\mathcal{J}}$*.*

where m *is the number of role inclusion axioms in* δ*.*

The definition of d-model now carries over from Definition 26, the only difference being that the new definition, $>_{KB,\delta}$ of the preference relation, is used when default role axioms are also included.

To show the decidability of reasoning, with any default-knowledge-base (KB, δ) having role defaults, we assume that KB is in a decidable DL $\mathcal{L}$ which supports nominal concept expression, boolean role constructors, concept products, and the universal role $\mathcal{U}$. In [81], it was shown that expressive DLs can be extended with boolean role constructors for simple roles without compromising on complexity and decidability. For some tuple $\mathcal{P} \equiv ((X_1, \ldots, X_n), (Y_1, \ldots, Y_m)) \in \mathcal{D}_{KB}$, let $KB_{\mathcal{P}}$ be the knowledge base that is obtained by adding the following axioms to KB. For each $C_i \sqsubseteq_d D_i \in \delta$, add the following:

1. $\overline{X_i} \equiv (C_i \sqcap D_i \sqcap \{a_1, \ldots, a_k\}) \sqcup (\neg C_i \sqcap \{a_1, \ldots, a_k\})$, where $\overline{X_i}$ is the nominal expression $\{x_1, \ldots, x_m\}$ containing exactly the named individuals in X_i, and $\{a_1, \ldots, a_k\} = \mathsf{Ind}_{KB}$.
2. $C_i \sqcap \neg\{a_1, \ldots, a_k\} \sqsubseteq D_i$, where $\{a_1, \ldots, a_k\} = \mathsf{Ind}_{KB}$.

And for each $R_j \sqsubseteq_d S_j \in \delta$, add the following:

1. For each $(a, b) \in Y_j$, add the ABox axiom $R_{a,b}(a, b)$ and the axiom

$$\{x\} \sqcap \exists R_{a,b}.\{y\} \sqsubseteq \{a\} \sqcap \exists \mathcal{U}.(\{y\} \sqcap \{b\}),$$

where $R_{a,b}$ is a fresh role name, and $\{x\}$ and $\{y\}$ are so-called *nominal schemas* as introduced in [58]: They are a kind of nominal variables, which can stand for any

nominal. In fact, this axiom can easily be cast into a set of axioms not containing nominal schemas, as shown in [58]. The axiom just given enforces that $R^{\mathcal{I}}_{a,b} \cap (\Delta^{\mathcal{I}}_{Ind} \times \Delta^{\mathcal{I}}_{Ind}) = \{(a, b)\}$.

2. $\bigsqcup_{(a,b)\in\mathcal{Y}_j} R_{a,b} \equiv (R_j \sqcap D_j \sqcap \mathcal{U}_g) \sqcup R_j \sqcap \neg\mathcal{U}_g$, where $\mathcal{U}_g \equiv \mathsf{Ind}_{\mathcal{KB}} \times \mathsf{Ind}_{\mathcal{KB}}$.

3. $R_j \sqcap \neg\mathcal{U}_g \equiv S_j$, where $\mathcal{U}_g = \mathsf{Ind}_{\mathcal{KB}} \times \mathsf{Ind}_{\mathcal{KB}}$.

The proceeding construction for role defaults is analogous to the one for class inclusion defaults, with the exception that we do not have a nominal constructor for roles. However, for the specific setting we have here, we can obtain the same result by using the axioms from points 1 and 2 just given.

It should also be noted that, the above outlined construction of $\mathcal{KB}_{\mathcal{P}}$ can be computed in a finite number of steps.

The remainder of the decidability argument for d-entailment now carries over easily from section 4.3, and we omit the details. It should be noted that the availability of boolean role constructors is required for our argument, and that corresponding simplicity restrictions may apply, depending on the concrete case.

4.5 Application of Defaults in Ontology Alignment

Variety and semantic heterogeneity are at the very core of many fields like the Semantic Web, and Big Data. To give a concrete example, many interesting scientific and societal questions cannot be answered from within one domain alone but span across disciplines, instead. Studying the impact of climate change, for instance, requires us to consider data and models from climatology, economics, biology, ecology, geography, and the medical science. While all these disciplines share an overlapping set of terms, the meanings of these terms clearly differ between disciplines. A street, for instance, is a *connection* between A and B from the view point of transportation science, and, at the same time, a disruptive

$$a{:}\mathit{flowsInto} \sqsubseteq a{:}\mathit{IsConnected} \tag{4.17}$$
$$a{:}\mathit{IrrigationCanal} \sqsubseteq a{:}\mathit{Canal} \tag{4.18}$$
$$\exists a{:}\mathit{flowsInto}.a{:}\mathit{AgriculturalField} \sqsubseteq a{:}\mathit{IrrigationCanal} \tag{4.19}$$
$$a{:}\mathit{Waterbody} \sqcap a{:}\mathit{Land} \sqsubseteq \bot \tag{4.20}$$
$$a{:}\mathit{AgriculturalField} \sqsubseteq a{:}\mathit{Land} \tag{4.21}$$
$$b{:}\mathit{flowsInto} \sqsubseteq b{:}\mathit{IsConnected} \tag{4.22}$$
$$b{:}\mathit{Canal} \sqsubseteq (\geq 2\ b{:}\mathit{IsConnected}.b{:}\mathit{Waterbody}) \tag{4.23}$$
$$b{:}\mathit{IrrigationCanal} \equiv (=1\ b{:}\mathit{isConnected}.b{:}\mathit{Waterbody}) \sqcap (=1\ b{:}\mathit{flowsInto}.b{:}\mathit{AgriculturalField}) \tag{4.24}$$

Figure 4.2: Fragments of two ontologies, (4.17)-(4.21), respectively (4.22)-(4.24), to be aligned.

separation which cuts through habitats from the view point of ecology. Even within single domains, the used terminology differs over time and space [47]. The idea that this variety should be 'resolved' is naive at best. The defaults extension proposed in this work can thus support more robust ontology alignments that respect variety and still allow us to share and integrate the heterogeneous data. In the following, we give some concrete examples that cannot be properly addressed with existing ontology alignment frameworks but would benefit from the proposed extension.

Consider the axioms in Figure 4.2. The ontology fragment consisting of (4.17) to (4.21) reflects a certain perspective on canals valid in a transportation application. In contrast, the axioms (4.22) to (4.24) reflect a different, but equally valid, perspective from an agricultural perspective. Typically, ontology alignment systems would default to a syntactic matching of shared primitives such as *AgriculturalField* or *IrrigationCanal*. However, applied to these two ontology fragments, this would yield a logical inconsistency in which some waterbodies would have to be land masses at the same time. Using our proposed free defaults, only certain canals from a would qualify as canals in b, avoiding the inconsistencies.

While in the above example the inconsistency was largely caused by the cardinality restrictions, other cases involve concrete domains. For instance, each US state (and the same argument can be made between counties as well) has its own legally binding definition of what distinguishes a town from a city. Thus, to query the Linked Data cloud for towns it is required to take these local definitions into account. Otherwise one would, among hundreds of thousands of small municipalities, also retrieve Los Angeles, CA or Stuttgart, Germany.[6] In several cases, these state-specific distinctions solely depend on the population count and, thus, could be handled using existing alignment systems. However, in other cases, they are driven by administrative divisions of geographic space, are based on historical reasons, or other properties. As argued before, our free defaults can handle these cases.

Let us now return to the example from section 4.1, and discuss it in more technical depth. We showed that an alignment using axiom (4.14) leads to inconsistency. Now consider instead using the approach of free defaults, by replacing axiom (4.14) with $b{:}hasSpouse \sqsubseteq_d a{:}hasSpouse$. As per our semantics, the pair $(b{:}mike, b{:}david)$ will act as an exception to the default role inclusion that we just added, and $a{:}hasSpouse(b{:}mike, b{:}david)$ will not hold anymore. On the other hand, if we also add the axiom $a{:}hasSpouse \sqsubseteq_d b{:}hasSpouse$ then $b{:}hasSpouse(a{:}john, a{:}mary)$ will also hold.

To see this formally, consider all the axioms of figure 4.1 except (4.14) to be $\mathcal{KB}$ and let $\delta \equiv \{(a{:}hasSpouse \sqsubseteq_d b{:}hasSpouse), (b{:}hasSpouse \sqsubseteq_d a{:}hasSpouse)\}$, and consider an interpretation $\mathcal{I}$, such that $(a{:}hasSpouse)^{\mathcal{I}} = \{(a{:}john^{\mathcal{I}}, a{:}mary^{\mathcal{I}})\}$ and $(b{:}hasSpouse)^{\mathcal{I}} = \{(b{:}mike^{\mathcal{I}}, b{:}david^{\mathcal{I}}), (a{:}john^{\mathcal{I}}, a{:}mary^{\mathcal{I}})\}$. Note that, forcing $(b{:}mike^{\mathcal{I}}, b{:}david^{\mathcal{I}})$ in the extension of $a{:}hasSpouse$ will result in an inconsistency because of the reasons mentioned in section 4.1. On the other hand, if we consider an interpretation $\mathcal{J}$, such that $(a{:}hasSpouse)^{\mathcal{J}} = \{(a{:}john^{\mathcal{J}}, a{:}mary^{\mathcal{J}})\}$ and $(b{:}hasSpouse)^{\mathcal{J}} = \{(b{:}mike^{\mathcal{J}}, b{:}david^{\mathcal{J}})\}$, then clearly $\mathcal{I} >_{\mathcal{KB},\delta} \mathcal{J}$, because the extension of $b{:}hasSpouse$ in $\mathcal{I}$ is greater than that of

[6] In case of DBpedia via `dbpedia:Stuttgart rdf:type dbpedia-owl:Town.`

$\mathcal{J}$. So, $\mathcal{I}$ is preferred over $\mathcal{J}$. In fact $\mathcal{I}$ is also a d-model of this default-knowledge-base.

The above example shows that in case of ontology alignments, using default constructs to map terms could help us avoid potentially inconsistent merged knowledge bases due to subtle semantic differences in the different ontologies.

As final example, we want to discuss the implications our approach has with respect to the use and abuse of owl:sameAs in linked data [37],[7] where this Web ontology language (OWL) construct is used extensively to provide links between different datasets. However, owl:sameAs is, semantically, a strong equality which equates entities, and this strong (specification-compliant) interpretation easily leads to complications. For instance, consider two linked datasets a and b where a contains the axioms: $\mathit{a{:}airport}(\mathit{a{:}kennedy})$ and $\mathit{a{:}airport} \sqsubseteq \mathit{a{:}place}$, and b contains axioms $\mathit{b{:}president}(\mathit{b{:}kennedy})$ and $\mathit{b{:}president} \sqsubseteq \mathit{b{:}person}$ plus the disjointness axiom $\mathit{b{:}person} \sqcap \mathit{a{:}place} \sqsubseteq \bot$.

Now, if some text-based co-reference resolution system identifies *a:kennedy* and *b:kennedy* as the same object, then it will result in a link such as $\mathit{owl{:}sameAs}\ (\mathit{a{:}kennedy}, \mathit{b{:}kennedy})$. Obviously, this yields to an inconsistency because of the disjointness axiom. However, if we use defaults this could be expressed as $\{\mathit{a{:}kennedy}\} \sqsubseteq_d \{\mathit{b{:}kennedy}\}$, which essentially is another way of saying that *a:kennedy* is identical to *b:kennedy unless* it causes an inconsistency. While [37] argues for a set of variants of equality with differing semantic strength, automatic identification of the exact variant of equality to be used is yet another matter, and presumably, rather difficult to accomplish. So for automated co-reference resolution, we would argue that the use of free defaults, which semantically recover from erroneous guesses by the alignment system, are a much more suitable solution.

[7] Note owl:sameAs is OWL representation of individual equality in DLs

4.6 Conclusion

In this chapter, we have provided a new semantics for embedding defaults into description logics and have shown that reasoning tasks are decidable in our setting. Both the decidable logic from [6] and our work are variants of Reiter's defaults [80]. But the approach in [6] is very restricted and arguably violates some key intuitions. Our proposal provides an improvement, mainly because with our free defaults, the application of defaults is not limited to named individuals. However, we impose that exceptions to the default rules only occur in the named individuals of the knowledge base. Also, our approach to the semantics is model-theoretic whereas most of the previous work on defaults has been mainly based on fixed point semantics [20, 25]. We have, furthermore, given a thorough account of the usefulness of free defaults in the context of ontology alignments. Through the examples in section 4.5, it is shown that the new semantics, which we have introduced in this chapter, is useful when dealing with the integration of heterogeneous ontologies. We believe that our work provides a foundation for a new and more powerful ontology alignment language.

Whether defaults over DLs are decidable when we allow exceptions to also occur over unnamed individuals, is still an open question and we intend to investigate this in the future. Future work also includes smart algorithmization and implementation of d-entailment, tasks mentioned in this chapter. A naive algorithm can easily be developed by searching for maximal d-model generating tuples from $\mathsf{P}^n(\mathsf{Ind}_{\mathcal{KB}})$, i.e. by searching for all maximal $\mathcal{P}$s in $\mathsf{P}^n(\mathsf{Ind}_{\mathcal{KB}})$ for which $\mathcal{KB}_{\mathcal{P}}$ has a classical model, and then using the process outlined in Theorem 7. Although this reasoning procedure appears to be decidable, it is very expensive and thus not feasible for practical use. However, the algorithmization could be made smarter by using some optimization techniques. For instance, $\mathsf{P}^n(\mathsf{Ind}_{\mathcal{KB}})$ could be represented as an ordered set of tuples where each tuple is a collection of comparable Ps sorted by the $\succ$ relation. The algorithm would then look for maximally satisfying Ps for each tuple by performing a binary search on every tuple. This should significantly improve

the performance of the naive approach since the number of steps to find all suitable $\mathcal{P}$s has been reduced by a large factor. These and other optimizations will be central to our investigation of algorithmizations.

5 Default Logic Based Ontology Alignment for Tractable DLs

In this chapter, we present a new semantics of a mapping language for the tractable fragments of description logics. We make use of default logic as the basis of the semantics of the new mapping language and show that this language is decidable.

5.1 Introduction

Description logic (DL) based knowledge representation is gaining in popularity, and with that, the number of domain ontologies is also on the rise. Especially in the medical domain, tractable fragments of DLs are heavily used. For example, SNOMED CT is a medical ontology which consists of more than 300,000 concepts, and which can be described in the description logic $\mathcal{EL}$ [4]. Smaller fragments of DLs are especially interesting for application scenarios where fast and efficient reasoning may be critical.

In this chapter, we provide a formal framework for dealing with defeasible reasoning for smaller fragments of DLs, especially in the context of ontology alignment. In particular we consider a language $\mathcal{ER}_{\bot,\mathcal{O}}$ which allows for conjunction, existentials, role chains, disjointness of concepts and ABox statements. We provide semantics for one-way (defeasible) alignments from terms in several ontologies to one overarching ontology, such that

queries can be asked in terms of this overarching ontology, while answers may contain instances from several lower level ontologies. For defeasibility we take motivation from default logic [80] and define the semantics along similar lines. It turns out that, combining DLs with default-like semantics is not very straightforward, as unrestricted default applications may result in undecidability [6, 83]. Previously, decidability was usually obtained for such logics by restricting defeasibility to known individuals, i.e. to a finite set of entities. In this chapter, we show that the combination of defeasible mappings with DLs presented here is decidable even without this type of restriction. Decidability in our setting results from the restriction to a tractable language in the $\mathcal{EL}$ family, together with the avoidance of recursion through the defeasible axioms resulting from our specific, but an important application scenario, namely the one-way alignment of ontologies.

Indeed, similar concepts appear in several ontologies from heterogeneous domains, but these concepts may slightly differ semantically. The motivation of using defeasible axioms as alignments stems from the need to handle such heterogeneity among various data models. As discussed in our previous work [83], DL axioms are semantically too rigid to be able to deal with alignments in such heterogeneous settings, in particular in light of the fact that ontology alignment systems mostly rely on string similarity matching [19]. For example, the concept that represents those human beings who consume only vegetarian food may be part of two different domain ontologies, but the notion of what vegetarian food means might slightly differ depending on the context, e.g. in some places eggs might be part of a typical vegetarian diet while in others this may not be so. Aligning these different world views appropriately cannot be done by simply mapping the respective concepts representing a "vegetarian person" in different ontologies, as claiming that they were equivalent may lead to inconsistencies.

For example, consider the axioms in Figure 5.1 (see section 5.2 for explanations of the notation). Axioms 5.1–5.4 represent one ontology and axioms 5.5–5.11 another ontology. An alignment system may give alignments similar to axioms 5.12–5.15. Since *romeo* is an

$$Veg \sqcap NonVeg \sqsubseteq \bot \tag{5.1}$$
$$\exists consumes.EggFood \sqsubseteq NonVeg \tag{5.2}$$
$$consumes \circ contains \sqsubseteq consumes \tag{5.3}$$
$$\{juliet\} \sqsubseteq Veg \tag{5.4}$$
$$\{romeo\} \sqsubseteq Eggetarian \tag{5.5}$$
$$Eggetarian \sqsubseteq Vegetarian \tag{5.6}$$
$$Eggetarian \sqsubseteq \exists eats.Egg \tag{5.7}$$
$$Eggetarian \sqcap NonVegetarian \sqsubseteq \bot \tag{5.8}$$
$$\{caesar\} \sqsubseteq Vegetarian \tag{5.9}$$
$$\{caesar\} \sqsubseteq NotEggetarian \tag{5.10}$$
$$NotEggetarian \sqcap Eggetarian \sqsubseteq \bot \tag{5.11}$$
$$Vegetarian \equiv Veg \tag{5.12}$$
$$NonVeg \equiv NonVegetarian \tag{5.13}$$
$$EggFood \equiv Egg \tag{5.14}$$
$$eats \sqsubseteq consumes \tag{5.15}$$

Figure 5.1: Example mapping with selected axioms.

Eggetarian (axiom 5.5), he is also a *Vegetarian* (axiom 5.6). Since every *Vegetarian* is also a *Veg* as per the mapping axiom 5.12, *romeo* is a *Veg*. From axioms 5.5, 5.7, 5.14, 5.15 and 5.2 we obtain that *romeo* is also a *NonVeg*. But *Veg* and *NonVeg* are disjoint classes, so this results in an inconsistency. But applying the same rules to *caesar* does not cause an inconsistency. The usual process of repairing alignments like this is to remove mappings that cause the inconsistency [50]. But we would then lose the conclusion that *caesar* is also a *Veg*. If we replace the mapping axioms with defeasible axioms as introduced below, then we could achieve this outcome where we carry over the similarities while respecting the differences.

The chapter is organized as follows. In section 5.2, we set the preliminaries by describing the language $\mathcal{ER}_{\bot,\mathcal{O}}$. The context of ontology mappings as well as the syntax and the semantics of defeasible mapping axioms along with the discussion on decidability is presented in section 5.3. Section 5.4 contains a description of the relation of the semantics of this approach with that of answer set programming for logic programs. Finally, we provide closing remarks in section 5.5.

5.2 The Description Logic $\mathcal{ER}_{\bot,\mathcal{O}}$

We consider the DL $\mathcal{ER}_{\bot,\mathcal{O}}$ (see [4] for further background). Let N_C be a set of *atomic concepts* (or *atomic classes*), let N_R be a set of *roles*, and let N_I be a set of *individuals*, which contains an element $\iota_{R,D}$ for each pair $(R, D) \in N_R \times N_C$. These $\iota_{R,D}$ are called *auxiliary* individuals. *Complex class expressions* (short, *complex classes*) in $\mathcal{ER}_{\bot,\mathcal{O}}$ are defined using the grammar

$$C ::= A \mid \top \mid \bot \mid C_1 \sqcap C_2 \mid \exists R.C \mid \{a\},$$

where $A \in N_C$, $R \in N_R$ and C_1, C_2, C are complex class expressions. Furthermore, a *nominal class* (short, *nominal*) is represented as $\{a\}$, where $a \in N_I$. A *TBox* in $\mathcal{ER}_{\bot,\mathcal{O}}$ is a set of *general class inclusion* (*GCI*) axioms of the form $C \sqsubseteq D$, where C and D are complex classes. $C \equiv D$ abbreviates two GCIs $C \sqsubseteq D$ and $D \sqsubseteq C$. An *RBox* in $\mathcal{ER}_{\bot,\mathcal{O}}$ is a set of *role inclusion* (*RI*) axioms of the form $R_1 \circ \cdots \circ R_n \sqsubseteq R$, where $R_1, \ldots, R_n, R \in N_R$. An *ABox* in $\mathcal{ER}_{\bot,\mathcal{O}}$ is a set of GCIs of the form $\{a\} \sqsubseteq C$ and $\{a\} \sqsubseteq \exists R.\{b\}$, where $\{a\}, \{b\}$ are nominals and C is a complex class.

An $\mathcal{ER}_{\bot,\mathcal{O}}$ *knowledge base* or *ontology* is a set of TBox, RBox and ABox statements, which furthermore, satisfy the condition that nominals occur only in ABox statements. This condition is a restriction of $\mathcal{ER}_{\bot,\mathcal{O}}$ as compared to, e.g., the allowed use of nominals in OWL 2 EL: While we allow for a full ABox, the TBox remains free of nominals. In particular, axioms such as $A \sqsubseteq \exists R.\{a\}$, with A an atomic or complex class other than a nominal, are not allowed.

An *initial* $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base is an $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base which does not contain any auxiliary individuals.

Example 6. *The following is an example of an (initial) $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base.*

Table 5.1: Semantics of the language $\mathcal{ER}_{\bot,\mathcal{O}}$

Concept	Semantics
$\top$	$\Delta^{\mathcal{I}}$
$\bot$	$\emptyset$
$\{a\}$	$\{a^{\mathcal{I}}\}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$\exists R.D$	$\{x \mid$ there exists some y with $(x, y) \in R^{\mathcal{I}}$ and $y \in D^{\mathcal{I}}\}$
$R_1 \circ R_2$	$R_1^{\mathcal{I}} \circ R_2^{\mathcal{I}}$

$$\textit{Bird} \sqsubseteq \textit{Fly} \qquad\qquad \textit{Penguin} \sqcap \textit{Fly} \sqsubseteq \bot$$

$$\textit{Penguin} \sqsubseteq \textit{Bird} \qquad\qquad \{\textit{tom}\} \sqsubseteq \exists \textit{hasPet}.\textit{Penguin}$$

Next, we describe the semantics of the language $\mathcal{ER}_{\bot,\mathcal{O}}$ using the notion of interpretation. An *interpretation* $\mathcal{I}$ of an $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base KB is a pair $(\Delta^{\mathcal{I}}, .^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set of elements called the *domain of interpretation* and $.^{\mathcal{I}}$ is the *interpretation function* that maps every individual in KB to an element of $\Delta^{\mathcal{I}}$, every concept in KB to a subset of $\Delta^{\mathcal{I}}$, and every role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Concept expressions are interpreted as shown in Table 5.1. An interpretation $\mathcal{I}$ is a *model* of an $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base KB if it satisfies all the TBox, RBox and ABox axioms, such that if $C \sqsubseteq D$ then $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, if $R \sqsubseteq S$ the $R^{\mathcal{I}} \sqsubseteq S^{\mathcal{I}}$, and $\{a\} \sqsubseteq C$ then $a^{\mathcal{I}} \in C^{\mathcal{I}}$, respectively.

It is well-known that any such knowledge base can be cast into normal form, as follows.

Definition 30. *An initial $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base is in normal form if it contains axioms of only the following forms, where $C, C_1, C_2, D \in N_C$, $R, R_1, R_2 \in N_R$ and $a, b \in N_I$*

$$\bar{A} \sqsubseteq C, C \sqsubseteq D \mapsto \bar{A} \sqsubseteq D \tag{5.16}$$
$$\bar{A} \sqsubseteq C_1, \bar{A} \sqsubseteq C_2, C_1 \sqcap C_2 \sqsubseteq D \mapsto \bar{A} \sqsubseteq D \tag{5.17}$$
$$\bar{A} \sqsubseteq C, C \sqsubseteq \exists R.D \mapsto \bar{A} \sqsubseteq \exists R.D \tag{5.18}$$
$$\bar{A} \sqsubseteq \exists R.\bar{B}, \bar{B} \sqsubseteq C, \exists R.C \sqsubseteq D \mapsto \bar{A} \sqsubseteq D \tag{5.19}$$
$$\bar{C} \sqsubseteq \exists R.\bar{D}, \bar{D} \sqsubseteq \bot \mapsto \bar{C} \sqsubseteq \bot \tag{5.20}$$
$$\bar{A} \sqsubseteq \exists R.\bar{B}, R \sqsubseteq S \mapsto \bar{A} \sqsubseteq \exists S.\bar{B} \tag{5.21}$$
$$\bar{A} \sqsubseteq \exists R_1.\bar{B}, \bar{B} \sqsubseteq \exists R_2.\bar{C}, R_1 \circ R_2 \sqsubseteq R \mapsto \bar{A} \sqsubseteq \exists R.\bar{C} \tag{5.22}$$

Figure 5.2: $\mathcal{ER}_{\bot,\mathcal{O}}$ completion rules. New axioms resulting from the rules are added to the existing axioms in *KB*. Symbols of the form $\bar{A}$ can be either a class name or a nominal class. We initialize comp(*KB*) with *KB* and $C \sqsubseteq C$, $\bot \sqsubseteq C$, $\bot \sqsubseteq \bot$ for all named classes $C \in N_C$.

$C \sqsubseteq D$	$C_1 \sqcap C_2 \sqsubseteq D$	$R_1 \circ R_2 \sqsubseteq R$
$\exists R.C \sqsubseteq D$	$C_1 \sqcap C_2 \sqsubseteq \bot$	$\{a\} \sqsubseteq C$
$C \sqsubseteq \exists R.D$	$R_1 \sqsubseteq R$	$\{a\} \sqsubseteq \exists R.\{b\}$

Theorem 8. *For every initial $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base KB there exists a knowledge base KB′ in normal form, such that KB $\models \bar{A} \sqsubseteq B$ if and only if KB′ $\models \bar{A} \sqsubseteq B$, where $\bar{A}$ is a class name or a nominal and B is a class name occurring in KB.*

Definition 31. *Given an initial $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base KB in normal form, we define the following:*

1. *Completion: the completion $comp(KB)$ of KB is obtained from KB by exhaustively applying the completion rules from Figure 5.2.*
2. *Clash: a completion $comp(KB)$ of KB contains a clash if $\{a\} \sqsubseteq \bot \in comp(KB)$, for some nominal class $\{a\}$.*

It is easily verified that, repeated applications of completion rules on an initial $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base produces only axioms which are also in normal form, with one exception: Axioms of the form $\{a\} \sqsubseteq \exists R.D$, with $R \in N_R$ and $D \in N_C$, can also appear.

It is straightforward to show that, $comp(KB)$ is well-defined and that the completion process has a polynomial time complexity. This and the soundness and completeness results below are adapted from [4].

Theorem 9. *(soundness and completeness) Let KB be an initial $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base in normal form. Then every model of KB is a model of $comp(KB)$. Furthermore, if $comp(KB)$ contains a clash then KB is inconsistent.*

Conversely, if A is an atomic class or a nominal and B is an atomic class such that $KB \models A \sqsubseteq B$, then $A \sqsubseteq B \in comp(KB)$. Furthermore, if KB is inconsistent then $comp(KB)$ contains a clash.

Proof. There are two cases to consider one in which $comp(KB)$ does not contain a clash and the other in which $comp(KB)$ contains a clash. We first consider the case when there is no clash. Let $\mathcal{I}$ be a model of KB. Hence, $\mathcal{I}$ satisfies all the axioms in KB. It suffices to show that if the Left hand side (LHS) of a completion rule holds under $\mathcal{I}$ then so does the right hand side then by induction on the rule application $\mathcal{I}$ is also a model of $comp(KB)$.

For rule 5.16, we have from the LHS $\mathcal{I} \models C \sqsubseteq D, \bar{A} \sqsubseteq C$. $\bar{A}$ can be either a concept or a nominal. If $\bar{A}$ is a concept then $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ and $A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ from which we get $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Therefore, $\mathcal{I} \models A \sqsubseteq D$. If $\bar{A}$ is a nominal $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ and $a^{\mathcal{I}} \in C^{\mathcal{I}}$ from which we get $a^{\mathcal{I}} \in C^{\mathcal{I}}$. Therefore, $\mathcal{I} \models \{a\} \sqsubseteq D$.

For rule 5.17, we have from the LHS $\mathcal{I} \models C_1 \sqcap C_2 \sqsubseteq D, \bar{A} \sqsubseteq C_1, \bar{A} \sqsubseteq C_2$, if $\bar{A}$ is a concept then $C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $A^{\mathcal{I}} \subseteq C_1^{\mathcal{I}}$, and $A^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$. Therefore, we have $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ and

consequently $\mathcal{I} \models A \sqsubseteq D$. If $\bar{A}$ is a nominal then $C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $a^{\mathcal{I}} \in C_1^{\mathcal{I}}$, and $a^{\mathcal{I}} \in C_2^{\mathcal{I}}$. Therefore, we have $a^{\mathcal{I}} \in D^{\mathcal{I}}$ and consequently $\mathcal{I} \models \{a\} \sqsubseteq D$.

For rule 5.18, we have from the LHS $\mathcal{I} \models C \sqsubseteq \exists R.D, \bar{A} \sqsubseteq C$. If $\bar{A}$ is a concept then, $C^{\mathcal{I}} \subseteq (\exists R.D)^{\mathcal{I}}, A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$. Therefore, we have $A^{\mathcal{I}} \subseteq (\exists R.D)^{\mathcal{I}}$ and consequently $\mathcal{I} \models A \sqsubseteq \exists R.D$. However, if $\bar{A}$ is a nominal then $C^{\mathcal{I}} \subseteq (\exists R.D)^{\mathcal{I}}$, $a^{\mathcal{I}} \in C^{\mathcal{I}}$. Therefore, we have $a^{\mathcal{I}} \in (\exists R.D)^{\mathcal{I}}$ and consequently $\mathcal{I} \models \{a\} \sqsubseteq \exists R.D$.

For rule 5.19, from the LHS we have $\mathcal{I} \models \exists R.C \sqsubseteq D, \bar{A} \sqsubseteq \exists R.\bar{B}, \bar{B} \sqsubseteq C$, then we have three possible cases. Both $\bar{A}, \bar{B}$ are concepts A, B respectively, then $B^{\mathcal{I}} \subseteq C^{\mathcal{I}}, A^{\mathcal{I}} \subseteq (\exists R.B)^{\mathcal{I}}, (\exists R.C)^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Consider, $A^{\mathcal{I}} \subseteq (\exists R.B)^{\mathcal{I}}$, which means $A^{\mathcal{I}} \subseteq \{x \in \Delta^{\mathcal{I}} \mid$ there exists a $y \in \Delta^{\mathcal{I}}$ with $(x, y) \in R^{\mathcal{I}}$ and $y \in B^{\mathcal{I}}\}$, and since $B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$, we get, $A^{\mathcal{I}} \subseteq \{x \in \Delta^{\mathcal{I}} \mid$ there exists a $y \in \Delta^{\mathcal{I}}$ with $(x, y) \in R^{\mathcal{I}}$ and $y \in C^{\mathcal{I}}\}$, which means $A^{\mathcal{I}} \subseteq (\exists R.C)^{\mathcal{I}}$. And since $(\exists R.C)^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, we get $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Therefore, $\mathcal{I} \models A \sqsubseteq D$. If $\bar{A}$ is a nominal $\{a\}$ and $\bar{B}$ is a concept B. The proof of this case is similar to that of the first case and can be obtained by replacing all occurrences of $A^{\mathcal{I}} \subseteq$ with $a^{\mathcal{I}} \in$ and $A \sqsubseteq$ with $\{a\} \sqsubseteq$. Finally, if $\bar{A}, \bar{B}$ are nominals $\{a\}, \{b\}$ respectively, we can reuse the arguments of the first case, where we replace all occurrences of $A^{\mathcal{I}} \subseteq$ with $a^{\mathcal{I}} \in$, $A \sqsubseteq$ with $\{a\} \sqsubseteq$, and all the occurrences of B with $\{b\}$ and $B^{\mathcal{I}}$ with $\{b^{\mathcal{I}}\}$.

For rule 5.20, from LHS we have $\bar{C} \sqsubseteq \exists R.\bar{D}, \bar{D} \sqsubseteq \bot$. If $\bar{C}, \bar{D}$ are concepts, C, D respectively, we get $C^{\mathcal{I}} = \{x \mid$ there exists a $y \in D^{\mathcal{I}}$, such that $(x, y) \in R^{\mathcal{I}}\}$, but $D^{\mathcal{I}} = \emptyset$. Hence, $C^{\mathcal{I}} = \emptyset$. The other two cases can be proved in a similar manner.

For rule 5.21, from LHS we have $\mathcal{I} \models R \sqsubseteq S$ and $\bar{A} \sqsubseteq \exists R.\bar{B}$; then again, we have three cases. $\bar{A}, \bar{B}$ are concepts A, B respectively. We have, $A^{\mathcal{I}} \subseteq (\exists R.B)^{\mathcal{I}}$, which means $A^{\mathcal{I}} \subseteq \{x \in \Delta^{\mathcal{I}} \mid$ there exists a $y \in \Delta^{\mathcal{I}}$ with $(x, y) \in R^{\mathcal{I}}$ and $y \in B^{\mathcal{I}}\}$, and since

$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$, we get, $A^{\mathcal{I}} \subseteq \{x \in \Delta^{\mathcal{I}} \mid$ there exists a $y \in \Delta^{\mathcal{I}}$ with $(x, y) \in S^{\mathcal{I}}$ and $y \in B^{\mathcal{I}}\}$, subsequently, $A^{\mathcal{I}} \subseteq (\exists S.B)^{\mathcal{I}}$. Therefore, $\mathcal{I} \models A \sqsubseteq \exists S.C$. The other two cases are similar to that of Rule - 5.19 and the proofs can be obtained in a similar manner as done for Rule - 5.19.

For rule 5.22, from LHS we have $\bar{A} \sqsubseteq \exists R_1.\bar{B}, \bar{B} \sqsubseteq \exists R_2\bar{C}, R_1 \circ R_2 \sqsubseteq R$. $\bar{A}, \bar{B}, \bar{C}$ are concepts A, B, C respectively, then $B^{\mathcal{I}} \subseteq \{v \in \Delta^{\mathcal{I}} \mid$ there exists a $w \in \Delta^{\mathcal{I}}$ such that $(v, w) \in R_2^{\mathcal{I}}$ and $w \in C^{\mathcal{I}}\}$ and $A^{\mathcal{I}} \subseteq \{u \in \Delta^{\mathcal{I}} \mid$ there exists a $v \in \Delta^{\mathcal{I}}$ such that $(u, v) \in R_1^{\mathcal{I}}$ and $v \in B^{\mathcal{I}}\}$. Therefore, $A^{\mathcal{I}} \subseteq \{u \in \Delta^{\mathcal{I}} \mid$ there exists a $v \in \Delta^{\mathcal{I}}$ such that $(u, v) \in R_1^{\mathcal{I}}$ and $v \in \{v \in \Delta^{\mathcal{I}} \mid$ there exists a $w \in \Delta^{\mathcal{I}}$ such that $(v, w) \in R_2^{\mathcal{I}}$ and $w \in C^{\mathcal{I}}\}\}$. And since, $R_1^{\mathcal{I}} \circ R_2^{\mathcal{I}} \subseteq R^{\mathcal{I}}$, we have, $A^{\mathcal{I}} \subseteq \{u \in \Delta^{\mathcal{I}} \mid$ there exists a $w \in \Delta^{\mathcal{I}}$ with $(u, w) \in R^{\mathcal{I}}$ and $w \in C^{\mathcal{I}}\}$, which means $A^{\mathcal{I}} \subseteq (\exists R.C)^{\mathcal{I}}$, therefore $\mathcal{I} \models A \sqsubseteq \exists R.C$.

Note, in the rest of the cases we can have a similar proof with adjustments similar to that in the proofs of the previous rules. We simply list down all the possibilities. $\bar{A}$ is a nominal $\{a\}$, and $\bar{B}, \bar{C}$ are concepts B, C, respectively. $\bar{A}, \bar{B}$ are nominals, $\{a\}, \{b\}$ respectively and $\bar{C}$ is a concept C. $\bar{A}, \bar{B}, \bar{C}$ are nominals $\{a\}, \{b\}, \{c\}$, respectively.

Now, if $comp(KB)$ contains a clash of the form $\{a\} \sqsubseteq \bot$, then, as shown above, any model of KB should also satisfy all the axioms of $comp(KB)$, therefore $KB \models \{a\} \sqsubseteq \bot$. KB is inconsistent.

This shows the first part of the theorem.

For the converse, consider an interpretation $\mathcal{I}$ of $comp(KB)$ as follows:

$$\Delta^{\mathcal{I}} = \{x_C \mid C \in N_C\} \cup \{x_{\{a\}} \mid \{a\} \in KB\}$$

$$A^{\mathcal{I}} = \begin{cases} \emptyset, & \text{if } A \sqsubseteq \bot \in comp(KB) \\ \{x_C \mid C \sqsubseteq A \in comp(KB)\} \cup \{x_{\{a\}} \mid \{a\} \sqsubseteq A \in comp(KB)\}, & \\ & \text{if } A \sqsubseteq \bot \notin comp(KB) \end{cases}$$

$$\{a\}^{\mathcal{I}} = \begin{cases} \emptyset, & \text{if } \{a\} \sqsubseteq \bot \in comp(KB) \\ \{x_{\{a\}}\}, & \text{if } \{a\} \sqsubseteq \bot \notin comp(KB) \end{cases}$$

$$R^{\mathcal{I}} = \{(x_C, x_D) \mid C \sqsubseteq \exists R.D \in comp(KB)\} \cup$$

$$\{(x_{\{a\}}, x_D) \mid \{a\} \sqsubseteq \exists R.D \in comp(KB)\} \cup$$

$$\{(x_{\{a\}}, x_{\{b\}}) \mid \{a\} \sqsubseteq \exists R.\{b\} \in comp(KB)\}$$

We now show that $\mathcal{I}$ is also a model of $comp(KB)$ whenever $comp(KB)$ is clash free, and thereby, also a model of $KB \subseteq comp(KB)$. Let $\alpha \in comp(KB)$. If α is of the form $C \sqsubseteq D$, $C \sqsubseteq \bot$, $\{a\} \sqsubseteq D$, $C \sqsubseteq \exists R.D$, $\{a\} \sqsubseteq \exists R.D$, $\{a\} \sqsubseteq \exists R.\{b\}$, there is nothing to show.

For α of the form $C_1 \sqcap C_2 \sqsubseteq D$, let $x_C \in C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$, then $C \sqsubseteq C_1 \in comp(KB)$, and $C \sqsubseteq C_2 \in comp(KB)$, and thus $C \sqsubseteq D \in comp(KB)$ by rule - 5.16. Hence, $x_C \in D$ as required. For α of the form $\exists R.A \sqsubseteq B$, let $(x_C, x_D) \in R^{\mathcal{I}}$ and $x_D \in A^{\mathcal{I}}$, then $D \sqsubseteq A \in comp(KB)$ and $C \sqsubseteq \exists R.D \in comp(KB)$ and by rule - 5.19 we have $C \sqsubseteq B$ and thus $x_C \in B^{\mathcal{I}}$ as required.

For α of the form $R \sqsubseteq S$, let $(x_C, x_D) \in R^{\mathcal{I}}$, $C \sqsubseteq \exists R.D \in comp(KB)$, and from

rule - 5.21, we have $C \sqsubseteq \exists S.D$, and thus $(x_C, x_D) \in S^{\mathcal{I}}$ as required. For α of the form $R_1 \circ R_2 \sqsubseteq R$, let $(x_C, x_D) \in R_1^{\mathcal{I}}, (x_D, x_E) \in R_2^{\mathcal{I}}$. Then $C \sqsubseteq \exists R_1.D \in comp(KB)$ and $D \sqsubseteq \exists R_2.E \in comp(KB)$, then by rule - 5.22, we get $C \sqsubseteq \exists R.E$, and thus $(x_C, x_E) \in R^{\mathcal{I}}$.

For α of the form $C_1 \sqcap C_2 \sqsubseteq \bot$, we prove by contradiction. Let $x_C \in C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$, then $C \sqsubseteq C_1, C \sqsubseteq C_2 \in comp(KB)$, Then from rule 5.17, we get $C \sqsubseteq \bot$ but as per definition of $\mathcal{I}$ we have $C^{\mathcal{I}} = \emptyset$.

Let $A \sqsubseteq B \notin comp(KB)$, then we know $x_A \in A^{\mathcal{I}}$ and $x_A \notin B^{\mathcal{I}}$ since $A \sqsubseteq B \notin comp(KB)$. Then $x_A \in B^{\mathcal{I}} \setminus A^{\mathcal{I}}$ and $\mathcal{I} \not\models A \sqsubseteq B$, therefore $KB \not\models A \sqsubseteq B$.

Let $\{a\} \sqsubseteq B \notin comp(KB)$, then we know $x_{\{a\}} \in \{a\}^{\mathcal{I}}$ and clearly $x_{\{a\}} \notin B^{\mathcal{I}}$ as $\{a\} \sqsubseteq B \notin comp(KB)$, thus $\mathcal{I} \not\models \{a\} \sqsubseteq B$ and $KB \not\models \{a\} \sqsubseteq B$.

Let $comp(KB)$ be clash free then there is no axiom of the form $\{a\} \sqsubseteq \bot$, then, as shown above $\mathcal{I}$ is a model of $comp(KB)$, KB, thus KB is consistent.

The claim is proven by contraposition. □

Note now that the $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base given in Example 6 is inconsistent.

Central to the proof of Theorem 9 is the following construction, which we will also use later in this chapter.

Given an $\mathcal{ER}_{\bot,\mathcal{O}}$ knowledge base KB, let $\mathcal{I} = \mathcal{I}(KB)$ be defined as the following

interpretation of $comp(KB)$:

$$\Delta^{\mathcal{I}} = \{x_{\{a\}}, x_C \mid C \text{ is a class name in } KB \text{ and } a \text{ is an individual in } KB\}$$

$$A^{\mathcal{I}} = \begin{cases} \emptyset, & \text{if } A \sqsubseteq \bot \in comp(KB) \\ \{x_C \mid C \sqsubseteq A \in comp(KB)\} \cup \{x_{\{a\}} \mid \{a\} \sqsubseteq A \in comp(KB)\}, & \\ & \text{if } A \sqsubseteq \bot \notin comp(KB) \end{cases}$$

$$\{a\}^{\mathcal{I}} = \begin{cases} \emptyset, & \text{if } \{a\} \sqsubseteq \bot \in comp(KB) \\ \{x_{\{a\}}\}, & \text{if } \{a\} \sqsubseteq \bot \notin comp(KB) \end{cases}$$

$$R^{\mathcal{I}} = \{(x_C, x_D) \mid C \sqsubseteq \exists R.D \in comp(KB)\} \cup$$

$$\{(x_{\{a\}}, x_D) \mid \{a\} \sqsubseteq \exists R.D \in comp(KB)\} \cup$$

$$\{(x_{\{a\}}, x_{\{b\}}) \mid \{a\} \sqsubseteq \exists R.\{b\} \in comp(KB)\}$$

The proof of Theorem 9 shows that $\mathcal{I}$ is a model of both $comp(KB)$ and KB.

5.3 Mapping Ontologies with $\mathcal{ER}_{\bot,\mathcal{O}}$-Defaults

We consider a rather specific but fundamentally important scenario, namely, the integration of ontology-based information by means of an overarching ontology, as laid out and applied e.g. in [51, 77] – see also the discussion of this in [83]. One of the central issues related to this type of information integration is how to obtain the mappings of the to-be-integrated ontologies to the overarching ontology, as the manual creation of these mappings is very costly for large ontologies.

However, methods for the automated creation of such mappings – commonly referred to as *ontology alignment* – are still rather crude [19, 45], and are therefore prone to lead to inconsistencies of the integrated ontologies, as discussed in section 5.1. In order to

deal with this, we introduce a defeasible mechanism to deal with such mappings. For simplicity of presentation, we consider only two ontologies, with one taking the role of the overarching ontology. The other ontology can be considered the disjoint union of the ontologies which are to be integrated.

The following notion is going to be central.

Definition 32. *(Defeasible Axiom) A* defeasible axiom *is of the form* $C \sqsubseteq_d D$ *or* $R \sqsubseteq_d S$*, where* C, D *are class names and* R, S *are roles.*

Intuitively speaking, our intention with defeasible axioms is the following: It shall function just like a class inclusion axiom, unless it causes an inconsistency; in which case it should not apply to individuals causing this inconsistency. In a sense, such defeasible axioms act as a type of semantic debugging of mappings: The semantics itself encodes the removal of inconsistencies. More specifically speaking, given a defeasible axiom $C \sqsubseteq_d D$, instances of C will also be instances of D, except those instances of C which cause an inconsistency when also an instance of D. Such Cs are usually known as exceptions. Of course, this intuitive understanding of defeasible axioms is not entirely straightforward to cast into formal semantics.[1] We will give such a formal semantics in section 5.3.1 below.

Definition 33. *(Mappings) Let* $\mathcal{O}_1, \mathcal{O}_2$ *be two consistent* $\mathcal{ER}_{\bot,\mathcal{O}}$ *knowledge bases. A (*defeasible*)* mapping *from* $\mathcal{O}1$ *to* $\mathcal{O}2$ *is a defeasible axiom with the left-hand side of the axiom a concept or role from* $\mathcal{O}_1$*, and the right-hand side a concept or role from* $\mathcal{O}_2$*.*

Note that, here we restrict the mappings to axioms involving roles and atomic classes. However, we do so without loss of generality as $C \sqsubseteq_d D$, for complex classes C, D, can be replaced by adding the axiom $C \sqsubseteq A$ to $\mathcal{O}1$ and the axiom $B \sqsubseteq D$ to $\mathcal{O}2$, where A and B are new concept names, and replacing $C \sqsubseteq_d D$ in δ by $A \sqsubseteq_d B$. Similarly, our

[1] Different ways how to do this lead to different non-monotonic logics. This is a well-studied subfield of artificial intelligence, from which we take inspiration.

approach encompasses the specific case of *ontology population*, where $\mathcal{O}1$ is empty and all mappings are of the form $\{a\} \sqsubseteq_d C$.

Definition 34. *(mapped-tuple) Let $\mathcal{O}_1, \mathcal{O}_2$ be two ontologies in $\mathcal{ER}_{\bot,\mathcal{O}}$ with δ the set of defeasible mappings from $\mathcal{O}_1$ to $\mathcal{O}_2$. Then the tuple $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ is called a* mapped-tuple.

5.3.1 Semantics and Decidability

Given a mapped-tuple $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, we define the formal semantics of the mappings following our intuitive reading as discussed above. Informally speaking, the semantics of $C \sqsubseteq_d D$ is similar to that of normal defaults as in Reiter's default logic [80]: if x is in C, then it can be assumed that x is also in D, unless it causes an inconsistency with respect to the current knowledge.

We define the semantics formally as follows: For each mapping axiom $C \sqsubseteq_d D$ in δ, we define a set Cand that represents the set of axioms that could be possibly added to the completion of $\mathcal{O}_2$ as a result of the mapping axiom.

$$\mathsf{Cand}(C \sqsubseteq_d D) = \{\{a\} \sqsubseteq D \mid \{a\} \sqsubseteq C \in comp(\mathcal{O}_1)\} \tag{5.23}$$

Furthermore, we define the set $\mathsf{Cand}^{\mathsf{n}}$ as the power set of Cand for each mapping axiom.

$$\mathsf{Cand}^{\mathsf{n}}(C \sqsubseteq_d D) = \{X \mid X \subseteq \mathsf{Cand}(C \sqsubseteq_d D)\} \tag{5.24}$$

Similarly, we define the corresponding sets $\mathsf{Cand}_{\mathcal{R}}$ and $\mathsf{Cand}^{\mathsf{n}}_{\mathcal{R}}$ for mapping axioms involv-

ing roles.

$$\mathsf{Cand}_{\mathcal{R}}(R \sqsubseteq_d S) = \{\{a\} \sqsubseteq \exists S.\{b\} \mid \{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_1)\} \tag{5.25}$$

$$\mathsf{Cand}^{\mathsf{n}}_{\mathcal{R}}(R \sqsubseteq_d S) = \{X \mid X \subseteq \mathsf{Cand}_{\mathcal{R}}(R \sqsubseteq_d S)\} \tag{5.26}$$

Note that a and b may be auxiliary individuals.

Definition 35. *(Mapped Ontology) Let $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ be a mapped-tuple. Define* selections *and the corresponding* mapped ontology *as follows:*

(i) For each mapping axiom of the form $C \sqsubseteq_d D \in \delta$, a selection *for $C \sqsubseteq_d D$ is any $\Sigma_{C \sqsubseteq_d D} \subseteq \mathsf{Cand}^{\mathsf{n}}(C \sqsubseteq_d D)$.*

(ii) For each mapping axiom of the form $R \sqsubseteq_d S \in \delta$, a selection *for $R \sqsubseteq_d S$ is any $\Sigma_{R \sqsubseteq_d S} \subseteq \mathsf{Cand}^{\mathsf{n}}_{\mathcal{R}}(R \sqsubseteq_d S)$*

(iii) Given selections for all mappings $\mu \in \delta$, we use Σ to denote their union $\Sigma = \bigcup_{\mu \in \delta} \Sigma_\mu$, and call Σ a selection *for the given mapped-tuple.*

(iv) $\mathcal{O}_2^{\Sigma} = comp(\mathcal{O}_2) \cup \bigcup_{X \in \Sigma} X$ is then called a mapped ontology.

Note that, each mapped-tuple $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ can give rise to only a finite number of corresponding mapped ontologies, and the number is bounded by $|\mathsf{Cand}^{\mathsf{n}}(C \sqsubseteq_d D)|^{|\delta_1|} \times |\mathsf{Cand}^{\mathsf{n}}_{\mathcal{R}}(R \sqsubseteq_d S)|^{|\delta_2|}$, where δ_1 (respectively, δ_2) is the set of class (respectively, role) mappings contained in δ.

Definition 36. *(Preferred Mapping) Let $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ be a mapped-tuple. Then for any two mapped ontologies $\mathcal{O}_2^{\Sigma^i}, \mathcal{O}_2^{\Sigma^j}$ we say $\mathcal{O}_2^{\Sigma^i} \succ \mathcal{O}_2^{\Sigma^j}$ or $\mathcal{O}_2^{\Sigma^i}$* is preferred over *$\mathcal{O}_2^{\Sigma^j}$, if all of the following hold:*

- *$\Sigma^i_\mu \supseteq \Sigma^j_\mu$, for all $\mu \in \delta$*

- $\Sigma^i_\mu \supset \Sigma^j_\mu$, *for some* $\mu \in \delta$

Note that μ *can be of the form* $C \sqsubseteq_d D$ *or* $R \sqsubseteq_d S$.

The notion of preferred mapping is used to identify the individuals to which the defeasible axioms maximally apply.

Definition 37. *(Mapped Completion and Mapped Entailment) Given a mapped-tuple* $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, *let* $\mathcal{O}_2^\Sigma$ *be a mapped ontology obtained from some selection* Σ. *Then the completion* $comp(\mathcal{O}_2^\Sigma)$ *obtained by exhaustively applying the rules in Figure 5.2 is said to be a* mapped completion *of* $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ *if* $\mathcal{O}_2^\Sigma$ *is consistent and there is no consistent mapped ontology* $\mathcal{O}_2^{\Sigma^i}$, *such that* $\mathcal{O}_2^{\Sigma^i} \succ \mathcal{O}_2^\Sigma$ *holds.*

Furthermore, let α *an axiom of the form* $\{a\} \sqsubseteq \{b\}$ *or* $\{a\} \sqsubseteq \exists R.\{b\}$. *Then* α *is* entailed *by* $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, *written* $(\mathcal{O}_1, \mathcal{O}_2, \delta) \models_d \alpha$, *if* $\alpha \in comp(\mathcal{O}_2^\Sigma)$ *for each mapped completion* $\mathcal{O}_2^\Sigma$ *of* $(\mathcal{O}_1, \mathcal{O}_2, \delta)$.

Lemma 8. *A mapped-tuple* $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ *always has a mapped completion.*

Proof. There are two conditions for obtaining a mapped completion $comp(\mathcal{O}_2^\Sigma)$: (1) $\mathcal{O}_2^\Sigma$ is consistent, and (2) $\mathcal{O}_2^\Sigma$ is maximal, with respect to $\succ$. It is clear that there is at least one Σ such that $\mathcal{O}_2^\Sigma$ is consistent, namely $\Sigma = \emptyset$. If this is the only Σ producing a consistent mapped ontology, then $comp(\mathcal{O}_2^\Sigma)$ is the corresponding mapped completion. Now, let $\mathcal{S}$ be the set of all selections which produce a consistent mapped ontology. We already know that $\mathcal{S}$ is finite, and so, the set of corresponding consistent mapped ontologies is also finite, and therefore, contains maximal elements with respect to the preference relation $\prec$. Each of these maximal elements is then a mapped completion of $(\mathcal{O}_1, \mathcal{O}_2, \delta)$. □

Theorem 10. *The problem of entailment checking for a mapped-tuple* $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ *is decidable.*

Proof. In order to check entailment, it suffices to obtain all the possible mapped completions as per definition 37. Since there is only a finite number of possible selections for $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ then, as argued in the proof of Lemma 8, there is only a finite number of corresponding mapped ontologies, and furthermore, we know that exhaustive application of the completion rules terminates. Hence the task is decidable. □

5.3.2 Applying Defeasible Mappings to Unknowns

So far, we have defined the semantics of defeasible mappings and a way to derive entailments. Using these mappings, queries can be asked in terms of concepts of the ontology which is being mapped to.

For instance, let the ontology $\mathcal{O}_1$ have axioms sdad

$$\{john\} \sqsubseteq USCitizen \qquad \{john\} \sqsubseteq Traveler$$

$$USCitizen \sqsubseteq \exists hasPassport.USPassport,$$

let the ontology $\mathcal{O}_2$ have axioms

$$Tourist \sqsubseteq \exists hasPP.Passport$$

$$\exists hasPP.AmericanPassport \sqsubseteq EuVisaNotRequired,$$

and let δ consist of the mappings

$$\textit{Traveler} \sqsubseteq_d \textit{Tourist} \qquad\qquad \textit{hasPassport} \sqsubseteq_d \textit{hasPP}$$

$$\textit{USPassport} \sqsubseteq_d \textit{AmericanPassport}.$$

We can then ask questions in terms of the concepts and roles of $\mathcal{O}_2$, like "list all the tourists," i.e., all instances that belong to the class *Tourist*, and we would get the answer *john*. But if we look carefully, we would also expect *john* as an instance of the class *EuVisaNotRequired*.

However, as per the semantics we have defined in the previous section, we would not be able to derive this conclusion. This is because the defeasible mappings do not apply to unknowns. In this case, the unknown in question is *john*'s *USPassport*. We address this issue by modifying the semantics in order to apply the mappings to unknowns as well.

First of all, recall that the set N_I already contains the auxiliary individuals ι_{RC} for every $R \in N_R$ and $C \in N_C$ – we have not yet made use of them, but we will do so now. In fact, we now modify the completion rules in Figure 5.2 by adding two additional rules as follows, and where $a \in N_I$, i.e. a may also be an auxiliary individual.

$$\{a\} \sqsubseteq \exists R.D \mapsto \{a\} \sqsubseteq \exists R.\{\iota_{RD}\} \tag{5.27}$$

$$\{a\} \sqsubseteq \exists R.D \mapsto \{\iota_{RD}\} \sqsubseteq D \tag{5.28}$$

Furthermore, we retain all the definitions from section 5.3.1 starting from Cand, $\mathsf{Cand}_{\mathcal{R}}$ but using the completion $comp_u(\mathcal{O}_1)$ obtained by applying the completion rules in Figure 5.2 in conjunction with the new rules when producing selections. We still use $comp$, the previous version without the new rules, for all other steps.

Returning to the example above, $comp_u(\mathcal{O}_1)$ now becomes

$$\{\mathit{john}\} \sqsubseteq \mathit{USCitizen} \qquad \{\mathit{john}\} \sqsubseteq \mathit{Traveler}$$
$$\mathit{USCitizen} \sqsubseteq \exists \mathit{hasPassport}.\mathit{USPassport} \qquad \{\iota_{hpp,usp}\} \sqsubseteq \mathit{USPassport}$$
$$\{\mathit{john}\} \sqsubseteq \exists \mathit{hasPassport}.\{\iota_{hpp,usp}\},$$

and from the mappings we obtain:

$$\mathcal{O}_2^\Sigma = comp(\mathcal{O}_2) \cup \{ \quad \{\mathit{john}\} \sqsubseteq \mathit{Tourist}, \quad \{\mathit{john}\} \sqsubseteq \exists \mathit{hasPP}.\{\iota_{hpp,usp}\},$$
$$\{\iota_{hpp,usp}\} \sqsubseteq \mathit{AmericanPassport}\} \quad \}.$$

Note, that this $\mathcal{O}_2^\Sigma$ is the only maximal mapped ontology. When we apply the completion rules of Figure 5.2 on $\mathcal{O}_2^\Sigma$, rule 5.19 will produce the axiom $\{\mathit{john}\} \sqsubseteq \mathit{EuVisaNotRequired}$.

We now show that, under this new version, default mappings behave just as ordinary mappings, provided no inconsistencies arise. This is, of course exactly, what we would like to obtain, i.e., the new semantics is conservative in this respect and "kicks in" only if needed due to inconsistencies.

Theorem 11. *Let $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ be a mapped-tuple, such that for any selection Σ, $\mathcal{O}_2^\Sigma$ is consistent. Let α be an $\mathcal{ER}_{\bot,\mathcal{O}}$ axiom of the form $\{a\} \sqsubseteq C$ or $\{a\} \sqsubseteq \exists R.\{b\}$, where a, b are named individuals from $\mathcal{O}_1$ and C, R are class names, respectively role names, from $\mathcal{O}_2$. Then $(\mathcal{O}_1, \mathcal{O}_2, \delta) \models \alpha$ if and only if $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \bar{\delta} \models \alpha$, where $\bar{\delta}$ is exactly the same as δ but with all $\sqsubseteq_d$ replaced by $\sqsubseteq$.*

Proof. In this case, there is only one relevant selection Σ, namely the full selection, since for every possible Σ, $\mathcal{O}_2^\Sigma$ is consistent.

Consider an interpretation $\mathcal{I} = \mathcal{I}(\mathcal{O}_2^\Sigma)$ of $\mathcal{O}_2^\Sigma$, defined as at the end of Section 5.2, and

recall that $\mathcal{I} \models \mathcal{O}_2^\Sigma$.

Let $\mathcal{I}'$ be an interpretation of $\mathcal{O}_1 \cup \mathcal{O}_2^\Sigma$ which extends $\mathcal{I}$, such that $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{I}} \cup \{x_C \mid C \in N_C^{\mathcal{O}_1}\}$, and for all $C \in N_C^{\mathcal{O}_1}$ and $R \in N_R^{\mathcal{O}_1}$, $C^{\mathcal{I}'}$ and $D^{\mathcal{I}'}$ are constructed from $comp_u(\mathcal{O}_1)$ exactly as it is done for $\mathcal{I}$ from $comp(\mathcal{O}_2^\Sigma)$. Then clearly $\mathcal{I}' \models \mathcal{O}_2^\Sigma \cup \mathcal{O}_1$. Furthermore, axioms of the form $\{a\} \sqsubseteq C$, $\{a\} \sqsubseteq \exists R.\{b\}$ where $a, b \in N_I^{\mathcal{O}_1}, C \in N_C^{\mathcal{O}_2}$ and $R \in N_R^{\mathcal{O}_2}$ are only produced from the axioms of $\mathcal{O}_2^\Sigma$.

Moreover, $\mathcal{I}' \models \mathcal{O}_1 \cup \mathcal{O}_2 \cup \bar{\delta}$ holds. To prove this, it suffices to show that $\mathcal{I}'$ satisfies all axioms $C \sqsubseteq D \in \bar{\delta}$ and $R \sqsubseteq S \in \bar{\delta}$, since we already know that $\mathcal{I}' \models \mathcal{O}_1 \cup \mathcal{O}_2$. And indeed, for every axiom $C \sqsubseteq D \in \bar{\delta}$ (which also means $C \sqsubseteq_d D \in \delta$), we know that if $\{a\} \sqsubseteq C \in comp_u(\mathcal{O}_1)$ then $\{a\} \sqsubseteq D \in \mathcal{O}_2^\Sigma$. Hence, by definition of $\mathcal{I}'$, $a \in C^{\mathcal{I}'} \cap D^{\mathcal{I}'}$. Similarly, for every axiom $R \sqsubseteq S \in \bar{\delta}$, we know that whenever $\{a\} \sqsubseteq \exists R.\{b\} \in comp_u(\mathcal{O}_1)$, we have $\{a\} \sqsubseteq \exists S.\{b\} \in \mathcal{O}_2^\Sigma$, and by definition of $\mathcal{I}'$, we obtain $(a, b) \in R^{\mathcal{I}'}, S^{\mathcal{I}'}$.

So now, in particular, if $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \bar{\delta} \models \alpha$ then $\mathcal{I}' \models \alpha$, and therefore $\mathcal{I} \models \alpha$, since α does not contain any class or role names from $\mathcal{O}_1$. By definition of $\mathcal{I}$, we then obtain $\alpha \in comp(\mathcal{O}_2^\Sigma)$ and consequently $(\mathcal{O}_1, \mathcal{O}_2, \delta) \models \alpha$ as required.

Conversely, consider an interpretation $\mathcal{I} = \mathcal{I}(\overline{\mathcal{O}})$ of $\overline{\mathcal{O}} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \bar{\delta}$ obtained as defined at the end of Section 5.2, and recall that $\mathcal{I} \models \overline{\mathcal{O}}$.

Now consider $\mathcal{O}' = comp_u(\mathcal{O}_1) \cup comp(\mathcal{O}_2) \cup \delta \cup \Sigma$ and note that $\mathcal{O}_2^\Sigma = comp(\mathcal{O}_2) \cup \Sigma \subseteq \mathcal{O}'$ and also that $\overline{\mathcal{O}} \subseteq \mathcal{O}'$. Let $\mathcal{I}' = \mathcal{I}(\mathcal{O}')$ be obtained as defined at the end of Section 5.2, and recall that $\mathcal{I}' \models \mathcal{O}'$. By construction, we also obtain $\mathcal{I}' \models \mathcal{O}_2^\Sigma$ and also that $\mathcal{I}'$ and $\mathcal{I}$ coincide on the signature of $\overline{O}$.

So now, in particular, if $(\mathcal{O}_1, \mathcal{O}_2, \delta) \models \alpha$, for α as in the statement of the theorem, then $\mathcal{I}' \models \alpha$, and therefore $\mathcal{I} \models \alpha$, and by definition of $\mathcal{I}$ we obtain $\alpha \in comp(\mathcal{O}_1 \cup \mathcal{O}_2 \cup \bar{\delta})$.

Consequently, $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \bar{\delta} \models \alpha$ as required. □

5.4 Relationship with Answer sets

The above semantics is inspired by Reiter's default logic, as already mentioned. Formally, we show that it is very closely related with the prominent answer set semantics from logic programming, which in turn, has a well-established relationship to Reiter's default logic. We first recall the definition of answer sets from [30], see [40] for exhaustive background reading.

Definition 38. *(Answer Sets) An* extended program *is a logic program that contains rules of the form* $L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n$ *where* $0 \leq m \leq n$ *and each* L_i *is a literal* A *or* $\neg A$*.* $\neg$ *denotes so-called* classical negation*, as opposed to* not*, which denotes* default negation*.*

For Π *an extended program that contains no variables and does not contain* not *, let* Lit *be the set of ground literals in the language of* Π*. The* answer set $\alpha(\Pi)$ *of* Π *is the smallest subset* S *of* Lit*, such that*

1. *for any rule* $L_0 \leftarrow L_1, \ldots, L_m \in \Pi$*, if* $L_1, \ldots L_2 \in S$*, then* $L_0 \in S$*, and*
2. *if* S *contains a pair of complementary literals, then* $S = Lit$*.*

For Π *a (general) extended program and* Lit *the set of all literals in the language of* Π*, define* Π^S*, for a set* $S \subseteq Lit$*, as the extended program obtained by deleting, from* Π*,*

1. *each rule that has some* $not\ L$ *in its body with* $L \in S$*, and*
2. *all expressions of the form* $not\ L$ *in the bodies of the remaining rules.*

Finally, S *is an* answer set *of* Π *if* $S = \alpha(\Pi^S)$*.*

Let $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ be a mapped-tuple. We now define an extended program $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$

Table 5.2: Rewriting of axioms to rules

	Axiom	Rule
1	$C \sqsubseteq D$	$D(x) \leftarrow C(x)$
2	$C \sqsubseteq \bot$	$\neg C(x) \leftarrow C(x)$
3	$\exists R.C \sqsubseteq D$	$D(x) \leftarrow R(x,y) \wedge C(y)$
4	$C_1 \sqcap C_2 \sqsubseteq D$	$D(x) \leftarrow C_1(x) \wedge C_2(x)$
5	$C_1 \sqcap C_2 \sqsubseteq \bot$	$\neg C_2(x) \leftarrow C_1(x), \neg C_1(x) \leftarrow C_2(x)$
6	$R_1 \sqsubseteq R$	$R(x,y) \leftarrow R_1(x,y)$
7	$R_1 \circ R_2 \sqsubseteq R$	$R(x,z) \leftarrow R_1(x,y) \wedge R_2(y,z)$
8	$\{a\} \sqsubseteq C$	$C(a) \leftarrow$
9	$\{a\} \sqsubseteq \exists R.\{b\}$	$R(a,b) \leftarrow$

as follows. For every axiom of the form $C \sqsubseteq_d D \in \delta$ and for all $\{a\} \sqsubseteq C \in comp(\mathcal{O}_1)$, we add rules of the following form to $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$:

$$C(a) \leftarrow \tag{5.29}$$

$$D(a) \leftarrow C(a), not\ \neg D(a) \tag{5.30}$$

For mapping axioms of the form $R \sqsubseteq_d S \in \delta$, we add the following rules:

$$R(a,b) \leftarrow \tag{5.31}$$

$$S(a,b) \leftarrow R(a,b), not\ \neg S(a,b) \tag{5.32}$$

Furthermore, we add to $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$ all possible groundings of the rules obtained by rewriting $comp(\mathcal{O}_2)$, as per the rules in Table 5.2, using all the individuals that occur in $\mathcal{O}_1, \mathcal{O}_2$.

It should be noted that we do not provide a transformation for axioms of the form $C \sqsubseteq \exists R.D$ in Table 5.2. This is because for representing defeasible axioms in logic programs we need the classical negation [30] and to represent axioms with existentials on the right hand side we require existential rules. Although a stable model semantics

$$\{a\} \sqsubseteq C \quad (5.33) \qquad C \sqsubseteq_d D \quad (5.35) \qquad D \sqcap E \sqsubseteq \bot \quad (5.37)$$
$$\{a\} \sqsubseteq B \quad (5.34) \qquad B \sqsubseteq_d E \quad (5.36) \qquad D \sqsubseteq F \quad (5.38)$$
$$E \sqsubseteq F \quad (5.39)$$

Figure 5.3: Example mapping

for existential rules has been defined in [63], it is not defined for extended programs with classical negation. Furthermore, it is not straightforward to extend the approach from [63] to extended programs. So we restrict ourselves to showing that our reduction works for the case when axioms of the form $C \sqsubseteq \exists R.D$ are not present. This is sufficient to show that our approach aligns well with the answer set semantics.

Example 7. *Consider the axioms listed in Figure 5.3 where axioms 5.33, and 5.34 are from $\mathcal{O}_1$, axioms 5.37, 5.38, and 5.39 are from $\mathcal{O}_2$ and the axioms 5.35, and 5.36 represent the set δ of defeasible mappings. The corresponding extended program $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$ is as follows.*

$$C(a) \leftarrow \qquad D(a) \leftarrow C(a) \wedge \mathit{not}\ \neg D(a) \qquad \neg E(a) \leftarrow D(a) \qquad F(a) \leftarrow D(a)$$
$$B(a) \leftarrow \qquad E(a) \leftarrow B(a) \wedge \mathit{not}\ \neg E(a) \qquad \neg D(a) \leftarrow E(a) \qquad F(a) \leftarrow E(a)$$

Note there are two answer sets, $S_1 = \{C(a), B(a), D(a), \neg E(a), F(a)\}$ and $S_2 = \{C(a), B(a), E(a), \neg B(a), F(a)\}$, for $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$.

Definition 39. *Let $\mathcal{O}_2^\Sigma$ be a mapped ontology for $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, and let $comp(\mathcal{O}_2^\Sigma)$ be a corresponding mapped completion. Then we define the* mapped answer set *$S(\mathcal{O}_2^\Sigma)$ to be*

the following set:

$$\{C(a) \mid C \sqsubseteq_d D \in \delta \text{ and } \{a\} \sqsubseteq C \in comp(\mathcal{O}_1)\} \cup$$

$$\{R(a,b) \mid R \sqsubseteq_d S \in \delta \text{ and } \{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_1) \cup$$

$$\{C(a) \mid \{a\} \sqsubseteq C \in comp(\mathcal{O}_2^\Sigma)\} \cup$$

$$\{\neg D(a) \mid C \sqsubseteq_d D \in \delta, \{a\} \sqsubseteq C \in comp(\mathcal{O}_1) \text{ and } \{a\} \sqsubseteq D \notin comp(\mathcal{O}_2^\Sigma)\} \cup$$

$$\{R(a,b) \mid \{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_2^\Sigma)\} \cup$$

$$\{\neg S(a,b) \mid R \sqsubseteq_d S \in \delta, \{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_1) \text{ and } \{a\} \sqsubseteq \exists S.\{b\} \notin comp(\mathcal{O}_2^\Sigma)\}$$

Lemma 9. *Let $\mathcal{O}_2^\Sigma$ be a mapped ontology for $(\mathcal{O}_1, \mathcal{O}_2, \delta)$, and let $comp(\mathcal{O}_2^\Sigma)$ be a corresponding mapped completion. Then the mapped answer set, $S(\mathcal{O}_2^\Sigma)$, is an answer set of $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$.*

Proof. We use $\overline{\Pi}$ as a short notation for $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$. We need to show that (1) $S(\mathcal{O}_2^\Sigma) \subseteq Lit$ (the set of all literals in $\overline{\Pi}$), (2) $S(\mathcal{O}_2^\Sigma) = \alpha(\overline{\Pi}^{S(\mathcal{O}_2^\Sigma)})$. Condition (1) trivially holds. For condition (2), we first assume that $\delta = \emptyset$ then $\overline{\Pi}^{S(\mathcal{O}_2^\Sigma)} = \overline{\Pi}$, since $\overline{\Pi}$ does not contain any not in the rules. For this case, all the rules of $\overline{\Pi}$ are constructed using the rules of Table 5.2, followed by grounding. We show that $S(\mathcal{O}_2^\Sigma) = \alpha(\overline{\Pi})$ by showing that $S(\mathcal{O}_2^\Sigma)$ satisfies the conditions of definition 38. We only need to show that $S(\mathcal{O}_2^\Sigma)$ satisfies condition (1), since $\mathcal{O}_2$ is consistent. For all rules in $\overline{\Pi}$ of the form $C(a) \rightarrow D(a)$ generated using the transformation 1 of table 5.2, $S(\mathcal{O}_2^\Sigma)$ satisfies the condition: if $C(a) \in S(\mathcal{O}_2^\Sigma)$ then $D(a) \in S(\mathcal{O}_2^\Sigma)$. $C(a) \in S(\mathcal{O}_2^\Sigma)$, whenever $\{a\} \sqsubseteq C \in comp(\mathcal{O}_2^\Sigma)$ and if $C \sqsubseteq D \in comp(\mathcal{O}_2^\Sigma)$ then $\{a\} \sqsubseteq D \in comp(\mathcal{O}_2^\Sigma)$, therefore, $D(a) \in S(\mathcal{O}_2^\Sigma)$ by definition of $S(\mathcal{O}_2^\Sigma)$.

For rules of the form $C(a) \rightarrow \neg C(a)$ in $\overline{\Pi}$ generated using transformation 2, $S(\mathcal{O}_2^\Sigma)$

trivially satisfies the condition if: $C(a) \in S(\mathcal{O}_2^\Sigma)$ then $\neg C(a) \in S(\mathcal{O}_2^\Sigma)$ as $\mathcal{O}_2^\Sigma$ is consistent, therefore, no axiom of the form $\{a\} \sqsubseteq \bot \in comp(\mathcal{O}_2^\Sigma)$.

For rules of the form $R(x,y) \wedge C(y) \rightarrow D(x)$ in $\overline{\Pi}$ generated using transformation 3, $S(\mathcal{O}_2^\Sigma)$ satisfies the condition if: $R(a,b), C(b) \in S(\mathcal{O}_2^\Sigma)$, then $D(a) \in S(\mathcal{O}_2^\Sigma)$. We have $\exists R.C \sqsubseteq D \in comp(\mathcal{O}_2^\Sigma)$ by definition of the transformation rule. $R(a,b), C(b) \in S(\mathcal{O}_2^\Sigma)$ whenever $\{a\} \sqsubseteq \exists R.\{b\}, \{b\} \sqsubseteq C \in comp(\mathcal{O}_2^\Sigma)$, respectively, by definition of $S(\mathcal{O}_2^\Sigma)$. Then $\{a\} \sqsubseteq D \in comp(\mathcal{O}_2^\Sigma)$ should hold by rule 5.19 of the completion rules of $\mathcal{ER}_{\bot,\mathcal{O}}$. Hence, $D(a) \in S(\mathcal{O}_2^\Sigma)$.

For rules of the form $C_1(a) \wedge C_2(a) \rightarrow D(a)$ in $\overline{\Pi}$ generated using transformation 4, $S(\mathcal{O}_2^\Sigma)$ satisfies the condition if: $C_1(a), C_2(a) \in S(\mathcal{O}_2^\Sigma)$, then $D(a) \in S(\mathcal{O}_2^\Sigma)$. This is because $C_1(a), C_2(a) \in S(\mathcal{O}_2^\Sigma)$ whenever $\{a\} \sqsubseteq C_1, \{a\} \sqsubseteq C_2 \in comp(\mathcal{O}_2^\Sigma)$, since $C_1 \sqcap C_2 \sqsubseteq D \in comp(\mathcal{O}_2^\Sigma)$, $\{a\} \in D$ should be in $comp(\mathcal{O}_2^\Sigma)$, therefore, $D(a) \in S(\mathcal{O}_2^\Sigma)$.

Again for rules in $\overline{\Pi}$ generated using transformation 5 are trivially satisfied by $S(\mathcal{O}_2^\Sigma)$ since $\mathcal{O}_2^\Sigma$ is consistent.

For rules of the form $R_1(a,b) \rightarrow R(a,b)$ in $\overline{\Pi}$ generated using transformation 6, $S(\mathcal{O}_2^\Sigma)$ satisfies the condition if: $R_1(a,b) \in S(\mathcal{O}_2^\Sigma)$ then $R(a,b) \in S(\mathcal{O}_2^\Sigma)$. $R_1(a,b) \in S(\mathcal{O}_2^\Sigma)$, whenever $\{a\} \sqsubseteq \exists R1.\{b\} \in comp(\mathcal{O}_2^\Sigma)$ and since, $R_1 \sqsubseteq R \in comp(\mathcal{O}_2^\Sigma)$, $\{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_2^\Sigma)$ should hold, therefore, $R(a,b) \in S(\mathcal{O}_2^\Sigma)$.

For rules of the form $R_1(a,b) \wedge R_2(b,c) \rightarrow R(a,c)$ in $\overline{\Pi}$ generated using transformation 7, $S(\mathcal{O}_2^\Sigma)$ satisfies the condition: if $R_1(a,b), R_2(b,c) \in S(\mathcal{O}_2^\Sigma)$ then $R(a,c) \in S(\mathcal{O}_2^\Sigma)$. $R_1(a,b), R_2(b,c) \in S(\mathcal{O}_2^\Sigma)$ whenever $\{a\} \sqsubseteq \exists R_1.\{b\}, \{b\} \sqsubseteq \exists R_2.\{c\} \in comp(\mathcal{O}_2^\Sigma)$ and since $R_1 \circ R_2 \sqsubseteq R \in comp(\mathcal{O}_2^\Sigma)$, we have $\{a\} \sqsubseteq \exists R.C \in comp(\mathcal{O}_2^\Sigma)$. Hence, $R(a,c) \in S(\mathcal{O}_2^\Sigma)$.

For rules in $\overline{\Pi}$ generated using transformations 8 and 9, it is easy to see that $S(\mathcal{O}_2^\Sigma)$ trivially satisfy these rules since it follows directly from the definition of $S(\mathcal{O}_2^\Sigma)$.

Now consider the case when $\delta \neq \emptyset$. The mappings in δ gives rise to rules in $\overline{\Pi}$ of the form of rules in 5.29, 5.30, 5.31 and 5.32. For rules of type 5.29, 5.31 there is nothing to show and $S(\mathcal{O}_2^\Sigma)$ satisfies such rules (facts) by definition of $S(\mathcal{O}_2^\Sigma)$. Now rules of type 5.30, 5.32 are the only rules that contain not. Now consider $\overline{\Pi}^{S(\mathcal{O}_2^\Sigma)}$ as per the transformations in definition 38. Rules of type 5.30 are either deleted or they are transformed to the form $C(a) \rightarrow D(a)$. $S(\mathcal{O}_2^\Sigma)$ satisfies rules of this form since $C(a) \in S(\mathcal{O}_2^\Sigma)$ if $C \sqsubseteq_d D \in \delta$ and $\{a\} \sqsubseteq C \in comp(\mathcal{O}_1)$. Note that, for such cases there are only two possibilities $D(a) \in S(\mathcal{O}_2^\Sigma)$ or $\neg D(a) \in S(\mathcal{O}_2^\Sigma)$. Clearly, $\neg D(a) \notin S(\mathcal{O}_2^\Sigma)$, otherwise, the rule would have been deleted. Hence, $D(a) \in S(\mathcal{O}_2^\Sigma)$. Similarly, rules of the form 5.32 are either deleted or are transformed into $R(a,b) \rightarrow S(a,b)$. $R(a,b) \in S(\mathcal{O}_2^\Sigma)$ holds if $R \sqsubseteq_d S \in \delta$ and $\{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_1)$. Clearly, $S(a,b) \in S(\mathcal{O}_2^\Sigma)$, otherwise, the rule would have been deleted. Note, the arguments made above for the case of $\delta = \emptyset$ still hold and thereby $S(\mathcal{O}_2^\Sigma) = \alpha(\overline{\Pi}^{S(\mathcal{O}_2^\Sigma)})$. It can also be seen from the definition of $S(\mathcal{O}_2^\Sigma)$, it is in fact the smallest subset of Lit satisfying these conditions. □

Lemma 10. *Let $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ be a mapped-tuple and let S be an answer set of $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta)$ = $\overline{\Pi}$. Then $S = S(\mathcal{O}_2^\Sigma)$ for some mapped ontology $\mathcal{O}_2^\Sigma$ of $(\mathcal{O}_1, \mathcal{O}_2, \delta)$.*

Proof. First we construct $\mathcal{O}_2^\Sigma$ from S. Let $\Sigma_{C \sqsubseteq_d D} = \{\{a\} \sqsubseteq D \mid D(a) \in S\}$ for each mapping axiom $C \sqsubseteq_d D \in \delta$ and $\Sigma_{R \sqsubseteq_d S} = \{\{a\} \sqsubseteq \exists S.\{b\} \mid R(a,b) \in S\}$ for each mapping axiom $R \sqsubseteq_d S \in \delta$. Let Σ be the collection of all such $\Sigma_{C \sqsubseteq_d D}$ and $\Sigma_{R \sqsubseteq_d S}$. Then $\mathcal{O}_2^\Sigma = comp(\mathcal{O}_2) \cup \{X \mid X \in \Sigma\}$ is a mapped ontology as per definition 35. It remains

to be shown that $comp(\mathcal{O}_2^{\Sigma})$ is a mapped completion. We assume $\mathcal{O}_2^{\Sigma}$ is not a mapped completion which means there is some $\mathcal{O}_2^{\Sigma^i}$ with $comp(\mathcal{O}_2^{\Sigma^i})$ a mapped completion and $\mathcal{O}_2^{\Sigma^i} \succ \mathcal{O}_2^{\Sigma}$. Therefore, for some mapping axiom $C \sqsubseteq_d D \in \delta$ or $R \sqsubseteq_d S \in \delta$ we have $\Sigma^i_{C \sqsubseteq_d D} \succ \Sigma_{C \sqsubseteq_d D}$ or $\Sigma^i_{R \sqsubseteq_d S} \succ \Sigma_{R \sqsubseteq_d S}$. For the case when $\Sigma^i_{C \sqsubseteq_d D} \succ \Sigma_{C \sqsubseteq_d D}$ then there is an axiom $\{a\} \sqsubseteq D$ with $\{a\} \sqsubseteq C \in comp(\mathcal{O}_1)$ and $\{a\} \sqsubseteq D \in \mathcal{O}_2^{\Sigma^i}$, $\{a\} \sqsubseteq D \notin \mathcal{O}_2^{\Sigma}$. For the case when $\Sigma^i_{R \sqsubseteq_d S} \succ \Sigma_{R \sqsubseteq_d S}$ there is an axiom $\{a\} \sqsubseteq \exists S.\{b\} \in \mathcal{O}_2^{\Sigma^i}$ with $\{a\} \sqsubseteq \exists R.\{b\} \in comp(\mathcal{O}_1)$ and $\{a\} \sqsubseteq \exists S.\{b\} \notin \mathcal{O}_2^{\Sigma}$. Also from lemma 9, $S(\mathcal{O}_2^{\Sigma^i})$ is an answer set of $\overline{\Pi}$ with $D(a) \in S(\mathcal{O}_2^{\Sigma^i})$ or $S(a, b) \in S(\mathcal{O}_2^{\Sigma^i})$. But that could not be the case since $S \subset S(\mathcal{O}_2^{\Sigma^i})$ is an answer set of $\overline{\Pi}$ and an answer set should be the minimal subset of Lit satisfying the rules of $\overline{\Pi}$ as per definition 38. □ □

The following theorem is now a direct consequence of Lemmas 9 and 10.

Theorem 12. *Let $(\mathcal{O}_1, \mathcal{O}_2, \delta)$ be a mapped-tuple. Then $(\mathcal{O}_1, \mathcal{O}_2, \delta) \models_d \{a\} \sqsubseteq C$ if, and only if, $\Pi(\mathcal{O}_1, \mathcal{O}_2, \delta) \models_S C(a)$, where $\models_S$ represents stable model entailment.*

5.5 Conclusion

In this chapter, we provided an extension for the description logic $\mathcal{ER}_{\perp,\mathcal{O}}$ with the ability to have defeasible mappings between ontologies. This work should be easily extendable to other logics in the $\mathcal{EL}$ family, provided soundness and completeness proofs can be obtained for the base logic along similar lines. We show a reduction from our semantics of defeasible mappings to that of answer set programming. This shows that the approach outlined here is very close to the original notion of defaults. Furthermore, the application of defaults is not limited to named individuals but also applies to unknowns that are implicitly referred to in the knowledge base due to existentials.

Of course, our resulting logic appears to be no longer tractable. However, it should be remarked that the application of monotonic semantics is completely impossible in the context of inconsistencies coming from the mappings, and repair approaches currently require human intervention and are generally employed at the level of axioms, rather than individuals. Some form of paraconsistent reasoning [65] may be a more efficient contender, but then, paraconsistent approaches such as [65] tend to miss many desired consequences.

As a part of future work, we consider a smart algorithmization for entailment checking that would perform with reasonable efficiency. One possible approach would be to find a method to generate rules that act as templates which could be used to check which selections used to create the mapped ontologies would lead to inconsistencies without actually running the completion algorithm on the mapped ontologies. We also plan to implement the algorithm and perform a detailed evaluation of its performance, with respect to time when compared to the monotonic extensions and also with respect to the quality of entailments obtained by defeasible mappings, compared to traditional alignments produced by automatic alignment systems. We could make use of data made available by the ontology alignment evaluation initiative [24, 27]. Good results would lead to a solid framework towards a robust mapping language for tractable ontology languages.

6 Implementation and Evaluation

6.1 Implementation

In this chapter, we present an implementation of a reasoner built to support the semantics presented in Chapter 5 and to be used as a proof of concept. For this purpose, we use one of the well-known OWL 2 reasoners called Hermit [32] and build a wrapper around it to simulate the semantics of default mappings. Hermit is a sound and complete reasoner for all profiles of OWL. ELK [52], a consequence based reasoner built specifically for efficient OWL EL reasoning, was also considered to be used for the implementation, however, it turned out that ELK does not support some of the OWL API [41] reasoning methods which are critical to the default mapping reasoning process discussed in this dissertation.

In Algorithm 1, we outline the procedure to find all the mapped ontologies, as defined in Definition 35, for two given ontologies $\mathcal{O}_1, \mathcal{O}_2$. The input to the algorithm consists of two ontologies $\mathcal{O}_1$, $\mathcal{O}_2$. The output of this algorithm is a set of all mapped ontologies, as per the semantics of default mappings. In the first part of the algorithm, we initialize the sets MappedOntology and Σ as empty sets. MappedOntology is the set in which we would collect all the candidate mapped ontologies. Next, a call to the subroutine, findAllMappings is made and the return value is assigned to M. We call LogMap [48], an ontology matching and repair system, to generate the mappings from $\mathcal{O}_1$ to $\mathcal{O}_2$. We only retain the mappings from $\mathcal{O}_1$ to $\mathcal{O}_2$ from the mappings returned by LogMap. E.g., if LogMap returns $C_{\mathcal{O}_1} \equiv$

Algorithm 1 Construct all mapped ontologies for two ontologies $\mathcal{O}_1, \mathcal{O}_2$.

Require: Ontology $\mathcal{O}_1$, $\mathcal{O}_2$

```
1: MappedOntology ← ∅
2: Σ ← ∅
3: M ← findAllMappings(O1, O2)
4: precomputeInferences(O1)
5: for all m ∈ M do
6:    if m.type = ObjectPropertyMapping then
7:       Σ ← Σ× pow(getCandidatePairs(O1, m.left))
8:    else if m.type = ClassMapping then
9:       Σ ← Σ × pow(getCandidateInstances(O1, m.left))
10:   end if
11: end for
12: for all Σi ∈ Σ do
13:   if isConsistent(O2 ∪ getAxioms(Σi, M)) then
14:      MappedOntology ← O2 ∪ getAxioms(Σi, M)
15:   end if
16: end for
17: return findMaximal(MappedOntology)
```

$C_{\mathcal{O}_2}$ as one of the mappings, we convert it to $C_{\mathcal{O}_1} \sqsubseteq C_{\mathcal{O}_2}$, where $C_{\mathcal{O}_1}, C_{\mathcal{O}_2}$ are concepts from ontologies $\mathcal{O}_1, \mathcal{O}_2$, respectively. Similarly, we reject mappings of the form $C_{\mathcal{O}_1} \sqsupseteq C_{\mathcal{O}_2}$. The subroutine findAllMappings does exactly what we just described: calling LogMap to obtain the mappings and retain only those mappings which map from $\mathcal{O}_1$ to $\mathcal{O}_2$.

In the next step of the algorithm, we call the subroutine precomputeInferences. The subroutine, precomupteInferences, takes an ontology as input and computes all logical consequences that can be obtained from the axioms in the input ontology. To implement this procedure, we use Hermit's implementation of the precomputeInferences() method.

In the first for loop, we go through all the mappings obtained from the call to the procedure, findAllMappings. At each iteration, we check the type of mapping (concept or role) and, based on the type of the mapping, we call the appropriate procedure to get the candidate instances or pairs of instances for the mapping. In line 7, we call the procedure, getCandidatePairs, which returns a set of pairs of individuals that satisfy the role on the left-hand side of the mapping axiom. Formally, let $R_{\mathcal{O}_1} \sqsubseteq R_{\mathcal{O}_2}$ be a mapping that is

being processed in one of the iterations of the for loop, then getCandidatePairs will return all the pairs of individuals (a, b), such that $\mathcal{O}_1 \models R_{\mathcal{O}_1}(a, b)$. Similarly, for a mapping $C_{\mathcal{O}_1} \sqsubseteq C_{\mathcal{O}_2}$, the procedure getCandidateInstances returns the set of all individuals a, such that $\mathcal{O}_1 \models C_{\mathcal{O}_1}(a)$. Furthermore, the procedure pow returns the powerset of a given input set. In lines 7, 9, the set Σ is assigned the set corresponding to the Cartesian product of Σ from previous iteration and the set of candidate instances (or pairs) obtained from the calls to pow(getCandidateInstances($\mathcal{O}_1$, m.left)) (or pow(getCandidatePairs($\mathcal{O}_1$, m.left)) for pairs). Note that m.left represents the term on the left-hand side of the mapping axiom m.

Now it may be clear by this point of the algorithm that we have obtained the set of all possible selections Σ, from which we can select the maximal selections to identify all the mapped ontologies.

In the second for loop, we iterate through all the selections in Σ and collect all the selections for which the resulting mapped ontologies are consistent. To check the consistency of the mapped ontology for a given selection, at each iteration of the for loop, we call the procedure isConsistent with input argument $\mathcal{O}_2 \cup$ (getAxioms(Σ_i, M)). The procedure isConsistent checks if the input ontology is consistent using the Hermit reasoner. Algorithm 2 outlines the steps followed by the getAxioms procedure.

In the final step of Algorithm 1, we filter the MappedOntology set to retain only those mapped ontologies that correspond to maximal selections. We use the procedure findMaximal for this purpose. This procedure identifies the mapped ontologies as described in Definition 35.

The getAxioms procedure as defined in Algorithm 2 takes, as input, a selection set Σ_i and the set M containing the mapping axioms. The algorithm goes through all the mappings and creates corresponding assertion axioms. m.right represents the right hand side of the mapping axiom. We assume that we can look-up the candidate sets from Σ_i corresponding to each mapping axiom in M. The functions ObjectPropertyAssertionAxiom and

Algorithm 2 Get all axioms from a selection Σ_i, M. Assuming, Σ_i, M are indexed sets such that $\Sigma_i[j]$ corresponds to the set of candidates for $M[j]$.

Require: Selection Σ_i, a set of mappings M

```
Axioms ← ∅
for all m ∈ M do
  if m.type = ObjectPropertyMapping then
    for all (a, b) ∈ Σ_i[M.indexOf(m)] do
      Axioms ← Axioms ∪ ObjectPropertyAssertionAxiom(m.right, a, b)
    end for
  else if m.type = ClassMapping then
    for all a ∈ Σ_i[M.indexOf(m)] do
      Axioms ← Axioms ∪ ClassAssertionAxiom(m.right, a)
    end for
  end if
end for
return Axioms
```

ClassAssertionAxiom are constructor methods that create role assertion and class assertion axioms, as per the OWL API definitions, respectively.

Now, we describe the procedure to get logical consequences from the default mappings. In Algorithm 3, we outline a simple procedure to find all instances of a class or all pair of instances that satisfy a role. The procedure takes, as input, a set of mapped ontologies which one can obtain from Algorithm 1 together with a class or a role expression, and returns, as output, a set $\mathsf{Output} = \{a \mid \mathcal{O} \models P(a) \text{ for all } \mathcal{O} \in \mathsf{MappedOntology}\}$ if P is a class expression, and $\mathsf{Output} = \{(a, b) \mid \mathcal{O} \models P(a, b) \text{ for all } \mathcal{O} \in \mathsf{MappedOntology}\}$ if P is a role expression.

Algorithm 3 Get inferred property and class assertions.

Require: A set of mapped ontologies MappedOntology, a class or role expression P

```
Output ← ∅
for all O ∈ MappedOntology do
  O ← O ∩ getInferredAssertions(O, P)
end for
return Output
```

The overall process of checking mapped entailments for two ontologies $\mathcal{O}_1, \mathcal{O}_2$ is to

call Algorithm 1 to find all the mapped ontologies followed by a call to Algorithm 3 to query for instances (or pairs of instances) of a class expression (or a role expression).

In the following section, we describe the experiments and the results of the evaluation of the algorithm we just described.

6.2 Experiments

6.2.1 Mapping Marriage Ontologies

In Chapter 4, Figure 4.1, we show two example ontologies that represent two different world views on marriage. It is mentioned in that chapter, that using an ontology matching system would result in mappings that equate all the syntactically matching roles and concepts. In this section, we show the applicability of our default semantics using the proof of concept system that we described in the previous section.

Comparison With Ontology Mapping Repair

For the purpose of testing this example, we created the ontologies using the ontology editor Protégé [92]. The ontologies are shown in the Appendix. Note, that there are some slight differences in the ontologies used in the experiments to that in Figure 4.1, specifically, ontology 1 in the experiments, corresponds to ontology b from Figure 4.1 and ontology 2 in the experiments, corresponds to ontology a from Figure 4.1. Furthermore, we add some class assertions and role assertions to ontology 1. The mappings obtained from the LogMap system are as follows:

1. $\mathcal{O}_1$:*Male* == $\mathcal{O}_2$:*Male*
2. $\mathcal{O}_1$:*Female* == $\mathcal{O}_2$:*Female*
3. $\mathcal{O}_1$:*hasSpouse* == $\mathcal{O}_2$:*hasSpouse*

Table 6.1: Results of Experiment 1

System	Query	Instances
Default	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(mark, julie),(john, mary), (jacob, jane)*}
Default	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john*}
Default	$\mathcal{O}_2$:*Female(?)*	{*mary, julie, jane*}
LogMap Repair	$\mathcal{O}_2$:*hasSpouse(?,?)*	Inconsistent ontology
LogMap Repair	$\mathcal{O}_2$:*Male(?)*	Inconsistent ontology
LogMap Repair	$\mathcal{O}_2$:*Female(?)*	Inconsistent ontology

LogMap returns mapping correspondences in terms of $==$, $<=$, $>=$ that represent $\equiv, \sqsubseteq, \sqsupseteq$, respectively. We ran through the procedure described in the previous section and outline the results in Table 6.1.

When using our system under the semantics of default logic based mappings, we get the answers to the query *hasSpouse(?,?)* as a set containing three pairs as shown in Table 6.1. Note, that only the pair (*john, mary*) is from ontology 2, whereas, the other pairs are obtained in the answer due to the mappings. The LogMap repair system did not give any answers, as it failed due to the inconsistency of the merged ontology. The LogMap system throws an exception when trying to repair the merged ontology.

In this experiment, we were able to verify the results that we were expecting from the default system. To get an idea of what could be the possible automatic repair options, we created a merged version of these two ontologies using the three mapping axioms and ran the explain functionality on Protégé using Hermit. The explain functionality provides justifications for an inconsistent ontology. Justifications are defined as a minimal set of axioms that, taken together, cause an inconsistency. The result of the explanation tool is shown in Figure 6.1, which shows that there are a total of 8 axioms, which, if taken together, cause the inconsistency in the merged ontology. An automatic mapping repair function would most probably remove one of the two mapping axioms from the justifications to repair the mapping. We analyze what would be the impact of removing the mapping axioms

Figure 6.1: Protégé explanation window for the inconsistency in the merged ontology.

Table 6.2: Answer to queries after removing the mapping, $\mathcal{O}_1 : Male \sqsubseteq \mathcal{O}_2 : Male$

Ontology	Query	Instances
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\delta_1\}$	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(mark, julie),(john, mary), (jacob, jane), (david, mike)*}
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\delta_1\}$	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john*}
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\delta_1\}$	$\mathcal{O}_2$:*Female(?)*	{*mary, julie, jane*}

one at a time.

Suppose the mapping repair system repairs the mapping by removing the axiom $\mathcal{O}_1 : Male \sqsubseteq \mathcal{O}_2 : Male$. To find the impact of this repair, we removed this mapping from the merged ontology and ran the Hermit reasoner to compute logical consequences. The results are shown in Table 6.2, where δ_1 is the modified set of mappings. It can be seen that the answer to query $\mathcal{O}_2$:*hasSpouse* returns four pairs including $(david, mike)$. This is an undesired consequence, as both *david* and *mike* are instances of *Male* in the lower-level ontology. However, as shown in Table 6.1, our system does not produce this as a logical consequence.

In the other case, we remove the mapping, $\mathcal{O}_1 : hasSpouse \sqsubseteq \mathcal{O}_2 : hasSpouse$ and examine the logical consequences of the resulting merged ontology $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \delta_2$. After running the reasoning process on this merged ontology, we get the logical consequences as shown in Table 6.3. In this case, we lose some desired logical consequences,

Table 6.3: Answer to queries after removing the mapping, $\mathcal{O}_1$: *hasSpouse* $\sqsubseteq$ $\mathcal{O}_2$: *hasSpouse*

Ontology	Query	Instances
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\delta_2\}$	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(john, mary)*}
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\delta_2\}$	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john, mike, david*}
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\delta_2\}$	$\mathcal{O}_2$:*Female(?)*	{*mary, julie, jane*}

Table 6.4: Answer to queries when mapping from $\mathcal{O}_1$ to $\mathcal{O}_2 \cup \{\mathcal{O}_2 : \textit{Male(mike)}\}$

Ontology/System	Query	Instances
Default	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(john, mary), (jacob,jane), (mark, julie)*}
Default	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john, mike*}
Default	$\mathcal{O}_2$:*Female(?)*	{*mary, julie, jane*}
$\overline{\mathcal{O}}$	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(john, mary), (jacob,jane), (mark, julie), (david, mike)*}
$\overline{\mathcal{O}}$	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john, mike*}
$\overline{\mathcal{O}}$	$\mathcal{O}_2$:*Female(?)*	{*david, mary, julie, jane*}

i.e. $\mathcal{O}_2$:*hasSpouse*(*mark, julie*) and $\mathcal{O}_2$:*hasSpouse*(*jacob, jane*), as the two ontologies agree on these two role assertions. On the other hand, our system shows that these two assertions hold.

From the axioms of the two ontologies, $\mathcal{O}_1$ and $\mathcal{O}_2$, we know that there is no agreement between the two ontologies about the following assertions: *hasSpouse(mike, david)*, *Male(mike)* and *Male(david)*, therefore, none of these should appear as logical consequences. However, on applying any of the two possible repairs, we get some of these undesired logical consequences and, at the same time, we lose of some of the desired logical consequences.

One can also consider the possibility that $\mathcal{O}_2$:*Male(mike)* is present in $\mathcal{O}_2$ even before mapping the ontologies $\mathcal{O}_1, \mathcal{O}_2$. It is not unusual for ontologies to contain overlapping individuals, especially if the individual is a well-known entity. In that case, if we apply the first repair, that we considered above, then $\mathcal{O}_2$:*Female(david)* would be a logical consequence of the resulting repaired and merged ontology. In Table 6.4, we show the comparison of logical consequences obtained when running our system on $\mathcal{O}_1$,

$\mathcal{O}_2 \cup \{\mathcal{O}_2 : \textit{Male(mike)}\}$ and applying logical reasoning to the repaired, merged ontology $\overline{\mathcal{O}} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \{\mathcal{O}_2 : \textit{Male(mike)}\} \cup \delta_1$. As it can be seen in this case as well, our system performs better than the traditional repair techniques in the quality of answers to the queries. The repair technique leads to undesired answers *Female(david)* and *hasSpouse(mike, david)*, whereas the default mapping system provides more meaningful answers.

Comparison With Paraconsistent Reasoning

In this section, a comparison with an implementation of paraconsistent reasoning is presented to show the results of an alternative approach to deal with inconsistency in the mapped ontologies. The paraconsistent system used in this experiment is in accordance with the translations provided in [65].[1] The paraconsistent system takes an input ontology $\mathcal{O}$ and translates it into an ontology $\mathcal{O}'$, such that $\mathcal{O}' \models \alpha$, if and only if, $\mathcal{O} \models_P \alpha$, where α is an axiom in the language of $\mathcal{O}$ and $\models_P$ represents entailment under the semantics of paraconsistent description logics [65].

In this experiment, we chose the input ontology $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \delta$, where $\mathcal{O}_1, \mathcal{O}_2$ are the marriage ontologies from previous experiment and $\delta = \{\mathcal{O}_1 : \textit{Male} \sqsubseteq \mathcal{O}_2 : \textit{Male}, \mathcal{O}_1 : \textit{Female} \sqsubseteq \mathcal{O}_2 : \textit{Female}, \mathcal{O}_1 : \textit{hasSpouse} \sqsubseteq \mathcal{O}_2 : \textit{hasSpouse}\}$. $\mathcal{O}'$ is the translation we obtained from the translation procedure of the paraconsistent algorithm. Next, we ran OWL reasoner Hermit to obtain the logical consequences with input ontology $\mathcal{O}'$ which are nothing but the logical consequences of $\mathcal{O}$ as per the semantics of paraconsistent description logics. Table 6.5, shows the results of running the paraconsistent algorithm on $\mathcal{O}$. In this case, we get an undesired consequence of *hasSpouse(david, mike)* because, as discussed above, the two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ do not have an agreement on the *hasSpouse* relationship for *david* and *mike*. Whereas, our approach does not provide *hasSpouse(david, mike)*

Table 6.5: Query result comparison: Default mapping vs Paraconsistent logic

Ontology/System	Query	Instances
Default	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(john, mary), (jacob,jane), (mark, julie)*}
Default	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john, mike*}
Default	$\mathcal{O}_2$:*Female(?)*	{*mary, julie, jane*}
Paraconsistent	$\mathcal{O}_2$:*hasSpouse(?,?)*	{*(john, mary), (jacob,jane), (mark, julie), (david, mike)*}
Paraconsistent	$\mathcal{O}_2$:*Male(?)*	{*mark, jacob, john, mike*}
Paraconsistent	$\mathcal{O}_2$:*Female(?)*	{*mary, julie, jane*}

as a logical consequence which is more in line with the descriptions of the two ontologies.

In the above experiments, we have shown with a simple, yet demonstrative, example that using default logic based mappings avoids the possibility of logical inconsistencies and, at the same time, provides the desired logical consequences.

6.2.2 Mapping Biomedical Ontologies

In this part of the experiments section, we want to show the applicability of defaults in real world ontologies. We picked two ontologies from the Ontology alignment evaluation initiative (OAEI) [24][2]. OAEI aims to evaluate the quality of automatic ontology alignment systems and provide analysis on the state-of-the-art systems. The two ontologies we chose are from the "Large Biomedical Ontologies" track from OAEI 2014. The ontologies used for this experiment are (1) A fragment of the National Cancer Institute Thesaurus (NCI) ontology with file name oaei2014_FMA_small_overlapping_nci.owl ($\mathcal{O}_1$), and (2) Foundational Model of Anatomy (FMA) ontology with file name oaei2014_NCI_small_overlapping_fma.owl ($\mathcal{O}_2$).[3]

These ontologies are relatively large and contain only TBox axioms. $\mathcal{O}_1$ consists of 3,696 classes, and $\mathcal{O}_2$ consists of 6,488 classes. In the first step of the experiment, we ob-

[1]We acknowledge the co-operation of Fredrick Maier (Institute for Artificial Intelligence, University of Georgia) in providing us with the code for the paraconsistent system.

[2]Link to website: http://oaei.ontologymatching.org/

[3]URL to download the ontologies: http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/2014/LargeBio_dataset_oaei2014.zip

tained mappings between the two ontologies using the LogMap matching system. LogMap returned 2,709 mappings between $\mathcal{O}_1$ and $\mathcal{O}_2$. After importing the merged ontology to Protégé, it was identified that the class *Visceral_Pleura* was unsatisfiable due to the mappings. Upon running the explain functionality on Protégé, six sets of justifications were found as shown in Figures 6.2, 6.3, 6.4.

Figure 6.2: Protégé explanation window with explanations # 1, 2 for biomedical ontologies.

The root cause of the issue is the difference in meanings of the classes *Pleura*, *Visceral_Pleura* in $\mathcal{O}_1$ and $\mathcal{O}_2$. As can be seen from the explanations, everything in *Visceral_Pleura*, and everything in *Pleura*, are linked to two disjoint classes *Thoracic_Cavity* and *Lung* with the same role. Furthermore, *Visceral_Pleura* is a sub class of *Pleura*. Specifically, the following two axioms seem to be the cause of concern as they appear in all of

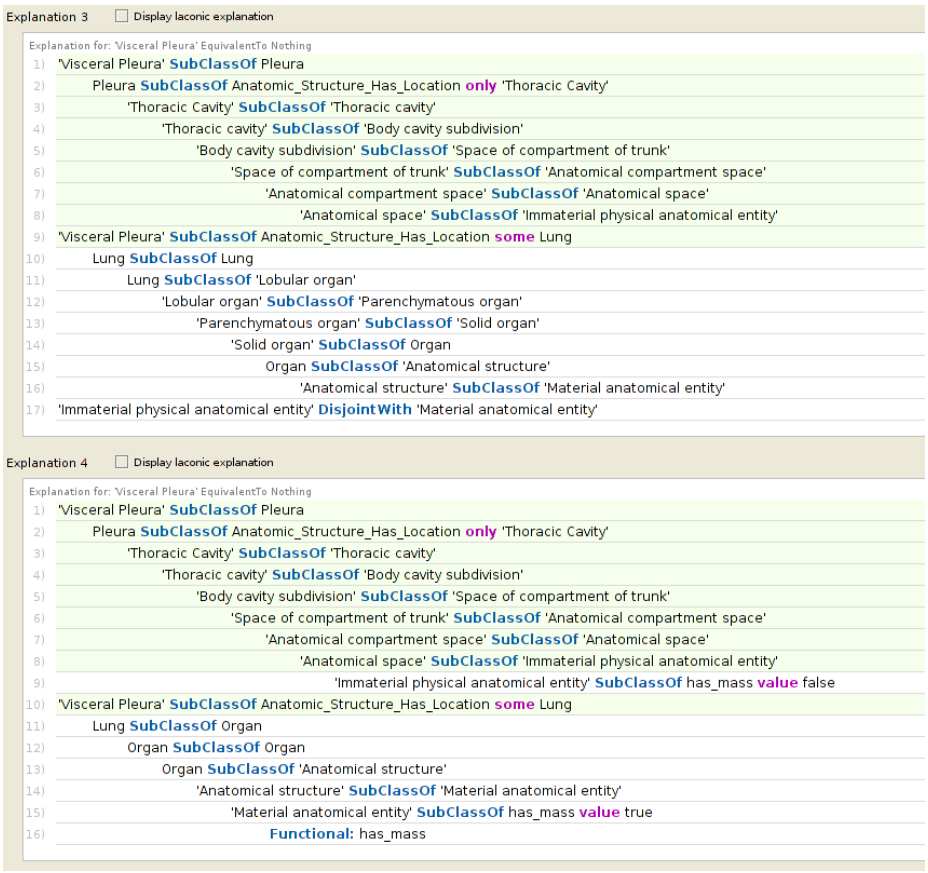

Figure 6.3: Protégé explanation window with explanations # 3, 4 for biomedical ontologies.

the justifications.

$$\mathcal{O}_1 : \mathit{Visceral\_Pleura} \sqsubseteq \exists \mathcal{O}_1 : \mathit{Anatomic\_Structure\_Has\_Location}.\mathcal{O}_1 : \mathit{Lung}$$

$$\mathcal{O}_1 : \mathit{Pleura} \sqsubseteq \forall \mathcal{O}_1 : \mathit{Anatomic\_Structure\_Has\_Location}.\mathcal{O}_1 : \mathit{Thoracic\_Cavity}$$

In $\mathcal{O}_2$, *Lung* and *Thoracic_Cavity* are mutually disjoint, whereas, these classes are not mutually disjoint in $\mathcal{O}_1$. An attempt to use LogMap repair functionality did not work as LogMap was unable to repair the mappings that are causing *Visceral_Pleura* to be unsatisfiable in the merged ontology. Repairing this problem manually is too complicated because of the complex relationships in both the ontologies. However, we tried removing the mapping axiom $\mathcal{O}_1$:*Thoracic_Cavity* $\sqsubseteq$ $\mathcal{O}_2$:*Thoracic_Cavity*, but the class *Visceral_Pleura*

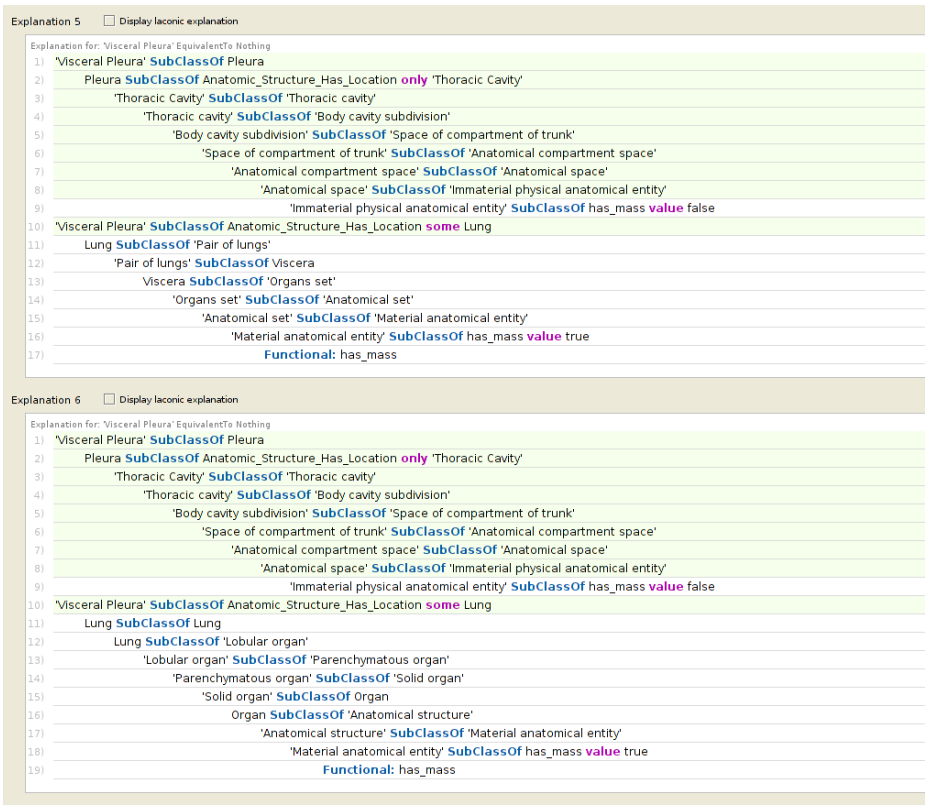

Figure 6.4: Protégé explanation window with explanations # 5, 6 for biomedical ontologies.

remained unsatisfiable in the merged ontology with a new set of justifications. Clearly, there are many disagreements among the two ontologies.

Note, that these ontologies do not contain any ABox, therefore, adding ABox to both the ontologies would lead to an inconsistency if *Visceral_Pleura* has at least one instance in any one of the ontologies.

We tested the working of our default mapping system for these ontologies. For that, we added some individuals to classes *Pleura*, *Visceral_Pleura*, *Thoracic_Cavity*, and *Lung* in $\mathcal{O}_1$ and ran our algorithm to generate mappings and find logical consequences from the mappings. Furthermore, we added the following two axioms in $\mathcal{O}_2$ to simulate the

Table 6.6: Results of Experiment 2

Concept	$\mathcal{O}_1$	$\mathcal{O}_2$ Inferred
Pleura	*Ind1001, Ind1003*	*Ind1001*
Visceral_Pleura	*Ind1003*	*None*
Thoracic_Cavity	*Ind1100, Ind1101*	*Ind1100, Ind1101*
Lung	*Ind1000*	*Ind1000*
Pair_Of_Lungs	*None*	*Ind1000*
Organ	*None*	*Ind1000*
Body_cavity_subdivision	*None*	*Ind1100, Ind1101*

inconsistency in the merged ontologies.

$$\mathcal{O}_2 : \textit{Visceral\_Pleura} \sqsubseteq \exists \mathcal{O}_2 : \textit{Anatomic\_Structure\_Has\_Location}.\mathcal{O}_2 : \textit{Lung}$$

$$\mathcal{O}_2 : \textit{Pleura} \sqsubseteq \forall \mathcal{O}_2 : \textit{Anatomic\_Structure\_Has\_Location}.\mathcal{O}_2 : \textit{Thoracic\_Cavity}$$

The default mappings used for this test case are as follows:

$$\mathcal{O}_1 : \textit{Visceral\_Pleura} \sqsubseteq_d \mathcal{O}_2 : \textit{Visceral\_Pleura}$$

$$\mathcal{O}_1 : \textit{Thoracic\_Cavity} \sqsubseteq_d \mathcal{O}_2 : \textit{Thoracic\_Cavity}$$

$$\mathcal{O}_1 : \textit{Pleura} \sqsubseteq_d \mathcal{O}_2 : \textit{Pleura}$$

Table 6.6, shows the result of running our algorithm to process the default mappings. The first column displays the concept name, the second column shows the instances added to $\mathcal{O}1$:*Concept*, and the third column represents the inferred instances in $\mathcal{O}_2$:*concept*. This exemplifies the use of default mappings, as the concept *Visceral_Pleura* differs in meaning in the two ontologies, only the instances of concept *Pleura* are carried over to that of ontology $\mathcal{O}_2$. Furthermore, we also get inferred instances for classes $\mathcal{O}_2$:*Thoracic_Cavity*, which would not be the case if we would remove the mapping

$\mathcal{O}_1$:*Thoracic_Cavity* $\sqsubseteq$ $\mathcal{O}_2$:*Thoracic_Cavity* and gather our conclusions from the merged ontology. The same holds for the class $\mathcal{O}_2$:*Body_cavity_subdivision*. We would not get *Ind1100* or *Ind1101* as logical consequences using the repair methods because of the axiom $\mathcal{O}_2$:*Thoracic_Cavity* $\sqsubseteq$ $\mathcal{O}_2$:*Body_cavity_subdivision*, and if, $\mathcal{O}_1$:*Thoracic_Cavity* $\sqsubseteq$ $\mathcal{O}_2$:*Thoracic_Cavity* is removed as a repair then *Ind1100, Ind1101* would not belong to the class $\mathcal{O}_2$:*Thoracic_Cavity*. This shows that using default mappings, we get more logical consequences whereas when we perform repair by removing axioms we lose many logical consequences, which is an undesirable result.

The above two experiments show the utility of the default based mapping language that we have presented in this dissertation. As it can be seen from both the synthetic as well as the real world ontology mapping cases, the use of defaults as a mapping language helps in avoiding inconsistencies that result due to the differences in meanings of same conceptual entities in different ontologies and at the same time provides the tools to infer the data about the similarity of the two ontologies.

7 Related Work

In this chapter, we present a coverage of the related work that is relevant to this dissertation. The related work is divided into two parts (1) Non-monotonic description logics, (2) Repairing ontology alignments.

7.1 Non-monotonic Description Logics

In this section, we cover the related work in the area of integrating description logics with non-monotonic features.

There are several approaches described in the literature for local closed world assumption (LCWA) which combine the open world assumption (OWA) and the closed world assumption (CWA) semantics, and in the following we briefly discuss some of the most important proposals which is relevant to the grounded circumscription approach in this dissertation.

Autoepistemic Logic [72, 73] is a thoroughly researched approach by a number of authors. The semantics of autoepistemic logic has been defined using an autoepistemic operator **K** [21, 22] and has been studied for $\mathcal{ALC}$ and also for more expressive DLs. [21, 23] further provide an epistemic operator **A** related to negation-as-failure which allows for the modeling of default rules and integrity constraints.

Circumscription [69] is another approach taken to develop LCWA extensions of DLs [11, 35, 36]. [11] evaluates the complexities of reasoning problems in variations of DLs

with circumscription. [35] provides examples to stress the importance of LCWA to provide an intuitive notion of matchmaking of resources in the context of Semantic Web Services. [36] provides an algorithmization for circumscriptive $\mathcal{ALCO}$ by introducing a preferential tableaux calculus, based on previous work on circumscription [10]. [53] proves a method to eliminate fixed predicates in circumscription patterns by adding negation of fixed predicates to the minimized set of predicates. A more recent work [8], contains a bigger coverage of complexity results for various fragments of DLs under the semantics of circumscription. An extension of grounded circumscription is presented in [7] which covers the reasoning procedure for integration of circumscription and the DL $\mathcal{SROIQ}$.

Some significant proposals involve the use of hybrid MKNF knowledge bases [44, 54, 75] which are based on an adaptation of the stable model semantics [30] to knowledge bases consisting of ontology axioms and rules, thereby combining both open world and closed world semantics. A variant of this approach using the well-founded semantics which has a lower computational complexity of reasoning has also been presented [55, 54], and the corresponding algorithms and implementations have been developed in [2, 33].

[25] takes a hybrid approach to combine ontologies and rules by keeping the semantics of both parts separate and at the same time allowing for building rules on top of ontologies and vice versa with some limitations, again following the Stable Model Semantics. [26] provides a related well-founded semantics.

Some of the work related to LCWA also involves the use of integrity constraints (ICs) and of the Unique Name Assumption (UNA). An approach extending OWL ontologies to add ICs such that it adds non-monotonicity to the DL is [74]. [90] provides semantics for OWL axioms to allow for IC and UNA to achieve local closed world reasoning. A recent approach [78] proposes the use of ICs using the semantics of grounded circumscription.

In [87], the notion of *DBox* is introduced. A DBox consists of a set of (atomic) assertions such that the extension of a DBox predicate under any interpretation is exactly as defined by this set of assertions. In a sense, grounded circumscription encompasses this

expressive feature but goes beyond it, while, as expected, loosing some of the desirable features of the more specialized DBox approach.

Related to the notion of free defaults, a series of work has recently been published which is related to typicality reasoning in description logics [31], in which the authors provide a minimal model semantics to achieve typicality. While our work is also based on preferred models, our goal is to follow some of the central ideas of default logic and to adapt it to provide a model-theoretic approach to defaults in DLs. Exact formal relationships between different proposals remain to be investigated. Other approaches that could be used to simulate defaults include circumscription [11, 84], while again the exact relationship between the approaches remains to be investigated. Also [9] talks about defeasible inclusions in the tractable fragments of DL, which again follows a similar intuition. We understand our proposal and results as a contribution to the ongoing discussion about the best ways to capture non-monotonic reasoning for Semantic Web purposes. Recent work [18, 16, 17] has been proposed in integrating the semantics of rational closure and KLM style semantics to DLs. These are alternative semantics to defaults and thus give a different perspective for apply defeasible logic to DLs.

7.2 Ontology Alignment Repair

The second part in this discussion of the related work involves the area of ontology alignment repair. As the main purpose of this research work is to provide a robust ontology mapping language, we take a look at other approaches that could be used to resolve the problematic alignments which result due the use of monotonic description logic (DL) constructs.

Paraconsistent description logics [62, 64, 65] make use of four-valued logics in order to deal with inconsistencies in knowledge bases. The semantics could be used to reason over merged ontologies even when the mappings cause an inconsistency under the mono-

tonic DL semantics.

Another approach which could be used to enable interchange of data using ontology mappings is the idea of distributed description logics (DDL) mainly proposed by [12, 85, 86]. DDL was designed to enable reasoning between multiple ontologies by using the so called bridge rules which are nothing but mappings using description logics axioms. This could be seen as an alternative approach to what we have proposed in this dissertation. This work also led to what is known as context OWL (C-OWL) [13] which provides the framework for DDL to contextualize ontologies. In [89], the authors propose how C-OWL could be used for alignment of medical ontologies.

With respect to repairing ontology alignments there are approaches like [60, 71]. Ontology mapping repair systems like Alcomo [70], ContentMap [49] and LogMap [48] have been developed that make use of justifications to identify possible repairs. A justification for a inconsistent ontology or a unsatisfiable concept is a minimal set of axioms which taken together cause the inconsistency. A repair is then to choose which of the axioms should be removed to perform repair. Therefore, the existing mapping repair systems remove one of the mapping axioms that belong to the set of justifications, whereas in our approach we partially apply the mappings to those individuals only that do not cause an inconsistency, thereby getting more logical consequences than the case of mapping repair.

The work in [71] is specifically close in spirit to our approach, though we provide a much more detailed semantic treatment which is closely related to Reiter's defaults and answer set programming. Our approach also forms a basis for a mapping language rather than focusing on the repairing of ontology alignments.

8 Conclusion

The objective of this work was to set the grounds for a robust ontology alignment language, such that applying the mappings do not render the ontologies useless. We provide a summary of the results obtained in this dissertation and point out some of the possible directions in which future work could extend/improve the results of this dissertation.

8.1 Summary

The contribution of this dissertation can be grouped into two broad categories: (1) Advancing the integration of non-monotonic logics with description logics, (2) Providing a basis for a new and powerful alignment language. The first part of this dissertation was to identify non-monotonic extensions of description logics that could be used as a basis for a robust ontology alignment language. In the second part, we identify a default based mapping language for a particular query answering scenario for tractable description logics. In the following sections, we summarize the contributions of this research work.

8.1.1 Grounded Circumscription (GC)

We provide a new semantics to circumscription which is more intuitive, as well as, has the property of decidability for all the reasoning tasks, even when roles are minimized. Furthermore, we also provide reasoning procedures to compute logical consequences of

knowledge bases using the semantics of GC. Indeed, the results shown in Chapter 3 improve upon previously known decidability results from [11], which showed that having closed roles with a non-empty TBox leads to undecidability. This was a severe limitation as decidability is the minimal requirement of any logic to be used in practice. We also argue in Chapter 3 that the semantics of GC is more intuitive than that of circumscription, as we restrict the extensions of the closed predicates to named individuals only.

In fact, the applicability of GC is visible by its use in applications like [78], where the authors propose an extension of integrity constraints based on GC and [67], as it makes use of GC closure semantics to model an ontology design pattern to address the issue of quantification over types.

8.1.2 Free Defaults

In Chapter 4, we introduced a new semantics for defaults when integrated with description logics. The decidability results shown in [6], essentially meant that defaults could only be used with description logics when the application of the default rules are restricted only to the named individuals in the knowledge base. However, in the case of defaults, it is rather counter-intuitive to not apply default rules to unknown individuals. In free defaults, we modified the semantics of defaults description logics by treating default rules as subsumption for unknown individuals, such that exceptions can occur only in named individuals, but defaults always apply to unknowns.

We showed that free defaults are decidable with the new, more intuitive semantics. Furthermore, we also showed decidability results for having role defaults which was previously not known. Various examples of application of free defaults is also shown in Chapter 4.

8.1.3 Mapping Language For $\mathcal{ER}_{\bot,\mathcal{O}}$

The mapping language introduced in Chapter 5 overcomes the limitations of integration of defaults with description logics. Wherein, we show that the application of defaults need not be restricted to named or unknown individuals. The scenario considered in this chapter is a framework with many, perhaps, heterogeneous ontologies at the lower-level and one over-arching ontology, such that, mappings from the lower-level ontologies are directional mappings to the over-arching ontology. This facilitates a query answering framework, such that queries can be framed using the vocabulary of the over-arching ontology and answers may contain individuals from the lower-level ontologies. Such a framework could also be used to understand the agreements and disagreements among the various lower level ontologies. This seems to be a very useful scenario given the heterogeneous nature of data descriptions across the ontologies over the web.

Apart from the usefulness of the $\mathcal{ER}_{\bot,\mathcal{O}}$ based mapping language, the most significant contribution of this work is to provide the first fragment of description logics, in which defaults could be integrated without the restrictions that were previously placed on the application of defaults in order to retain decidability. In fact, the proposed approach could be adapted to any of the tractable fragments of description logics which are commonly known as the $\mathcal{EL}$ family of logics. This research work opens the doors for further investigations on the fragments of description logics where defaults could be applied in an unrestricted manner. This leaves us with an open question to be answered: Is the unrestricted integration of defaults with non-tractable fragments of description logics truly undecidable?

Besides the theoretical foundations of a mapping language, we also implemented a proof of concept system to evaluate the utility of the proposed framework in real world applications. The experiments show promising results as we evaluate two mapping scenarios. In one, we show how our system behaves in practice when we try to map the example ontologies that present different world views on marital relationships. Indeed, the system

performs as expected by avoiding inconsistencies due to the mapping but at the same time provides useful answers to the queries. In the second experiment, we take two real world ontologies and again exemplify the utility of our approach to map ontologies.

8.2 Future Work

There are many directions in which future work could be pursued, extending the contributions of this dissertation. Indeed, there are some theoretical as well as some practical research questions that can be worked on in the future.

From the theoretical point of view, a detailed computational complexity analysis of the approaches presented in this dissertation would definitely be of worth to the researchers working in the area of the semantic web. Tight complexity bounds provide a good insight on the practical utility of the approaches. Therefore, establishing the complexity classes for our approaches, namely, grounded circumscription , free defaults and the default logic based mapping language would already be a substantial future work. For grounded circumscription and free defaults, we can also find the complexity of reasoning tasks for all the major fragments of DLs. We refer the interested reader to [11] as an inspiration for complexity analysis of a related approach.

Scalability is a major challenge when it comes to reasoning algorithms for the non-monotonic extensions with description logics. All the approaches discussed in this dissertation have a common problem, i.e., identifying the sets of maximal/minimal individuals in order to perform the reasoning tasks. In the worst case, we may end up checking all possible combinations and compare each other for maximality/minimality. The complexity of the brute force method to find all the maximal/minimal models of the knowledge base, would be at least exponential to the size of the set of all the named individuals in the knowledge base. Therefore, finding efficient procedures to perform the reasoning tasks of these approaches would be of prime importance.

The approach of summarizing ABox [28] could be adopted to produce efficient algorithms for our non-monotonic extensions of DLs. The idea of summarization, is to compress the size of the set of all the instances in the knowledge base by replacing it with a much smaller set of representative individuals. This results in a considerable amount of compression in the ABox. As discussed earlier, the larger the size of the ABox, the reasoning tasks become more time intensive. Therefore, shrinking the ABox would be useful in practice. The algorithms presented in [6], could also be used as inspiration to find the minimal/maximal models for the non-monotonic extensions. In [6], the authors present an algorithm to identify minimal inconsistent and maximal consistent sets of individuals. They employ a modified version of the tableaux procedure of $\mathcal{ALC}$ to compute the maximal consistent ABox to identify extensions of terminological default theories. The algorithm presented in this paper could, in fact, be adopted to implement an efficient procedure for grounded circumscription as well as for free defaults. It would be even more interesting to analyze the performance when we combine the summarization techniques and the maximal consistency algorithms to solve the reasoning problems for grounded circumscription and free defaults.

For the default logic based mapping language, algorithmic improvements could be achieved by understanding the sets of axioms that are mutually in conflict. For example, consider that we have two default mapping axioms, $C \sqsubseteq_d D$ and $A \sqsubseteq_d B$, for ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. Now, if $\mathcal{O}_2 \models D \sqcap B \sqsubseteq \bot$, then the same individual a cannot be carried over through both the defaults at the same time, as it would cause an inconsistency. In other words $(\mathcal{O}_1, \mathcal{O}_2, \delta) \not\models D(a)$ and $(\mathcal{O}_1, \mathcal{O}_2, \delta) \not\models B(a)$. We would then consider these two mapping axioms to be in mutual conflict. Therefore, if we could possibly identify the set of all mutually conflicting axioms as a pre-processing step, it would allow us to eliminate a lot of combinations without actually testing for inconsistency, thereby, improving the overall performance.

Of course, a solid implementation of the efficient algorithms could also be a part of

further future work. The implementation would allow the use of the approaches discussed in this dissertation for practical applications.

In this dissertation, we provided a default logic based mapping language, which does not place any restrictions on the application of the defaults to named or unknown individuals. We do so for the tractable fragments of DLs, however, with the restriction of one-way mappings from a set of ontologies to one over-arching ontology. The results of this work could be extended by investigating the decidability of this approach when the restriction of one-way mappings is lifted. The results would provide a significant understanding of the extent to which default logic can be integrated with DLs.

We have shown in this work, that there are still a lot of questions to be answered with regards to the integration of default logic with DLs. Indeed, we showed in the results of free defaults that there is scope in developing less restrictive approaches to deal with default logic in DLs. The major research question that needs to be looked at, is finding the maximal combination of DLs and default logic, which is decidable. Indeed, the answer to this question at the moment is that we do not know. The complications that arise when trying to prove the decidability/undecidability of the integration of default logic with DLs are the interplay between default rules and dealing with unknowns. We were able to show decidability results for the tractable fragments of DLs, as they have the property that lets us reuse the unknowns as role fillers for existentials on the right hand side of GCIs, without losing any logical consequences. However, this is not the case for expressive DLs, therefore, proving the decidability/undecidability is a rather complicated task.

There is also a possibility of investigating the relationship of the default logic integrations presented here with other related approaches that deal with defeasibility in description logics, including [31, 18, 16, 17]. The analysis could include a thorough comparison of the logical consequences obtained by these approaches, this would lead to a better understanding of relationships among these approaches.

The aim of this research work, was to provide a way to facilitate data interchange of

heterogeneous data sources. We provided, a default logic based mapping language that takes an alternative approach than the traditional ontology mappings. Instead of using description logic axioms to map and merge the ontologies, we make use of mappings as default rules that provide a bridge between various data sources. We carry over individuals from one knowledge base to other using these rules, such that inconsistency is avoided and, at the same time, sensible logical consequences are derived. We believe, that this lays a solid foundation for future research work in the area and the above remarks provide interesting and valuable research tasks to pursue.

Bibliography

[1] José-Luis Aguirre, Kai Eckert, Jérôme Euzenat, Alfio Ferrara, Willem Robert van Hage, Laura Hollink, Christian Meilicke, Andriy Nikolov, Dominique Ritze, François Scharffe, Pavel Shvaiko, Ondrej Sváb-Zamazal, Cássia Trojahn dos Santos, Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, and Benjamin Zapilko. Results of the Ontology Alignment Evaluation Initiative 2012. In Pavel Shvaiko, Jérôme Euzenat, Anastasios Kementsietsidis, Ming Mao, Natasha Fridman Noy, and Heiner Stuckenschmidt, editors, *Proceedings of the 7th International Workshop on Ontology Matching, Boston, MA, USA, November 11, 2012*, volume 946 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[2] José Júlio Alferes, Matthias Knorr, and Terrance Swift. Queries to Hybrid MKNF Knowledge Bases through Oracular Tabling. In Abraham Bernstein, David R. Karger, Tom Heath, Lee Feigenbaum, Diana Maynard, Enrico Motta, and Krishnaprasad Thirunarayan, editors, *Proceedings of the 8th International Semantic Web Conference*, ISWC '09, pages 1–16. Springer-Verlag, 2009.

[3] Dean Allemang and James A. Hendler. *Semantic Web for the Working Ontologist - Effective Modeling in RDFS and OWL, Second Edition*. Morgan Kaufmann, 2011.

[4] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the EL Envelope. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI-05, Proceedings of*

the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005, pages 364–369, 2005.

[5] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2nd edition, 2007.

[6] Franz Baader and Bernhard Hollunder. Embedding Defaults into Terminological Knowledge Representation Formalisms. *Journal of Automated Reasoning*, 14(1):149–180, 1995.

[7] Arjun Bhardwaj and Sangeetha. GC-SROIQ(C) : Expressive Constraint Modelling and Grounded Circumscription for SROIQ. *CoRR*, abs/1411.0406, 2014.

[8] Piero Bonatti, Marco Faella, Carsten Lutz, Luigi Sauro, and Frank Wolter. Decidability of Circumscribed Description Logics Revisited. In Thomas Eiter, Hannes Strass, Mirosaw Truszczyski, and Stefan Woltran, editors, *Advances in Knowledge Representation, Logic Programming, and Abstract Argumentation*, volume 9060 of *Lecture Notes in Computer Science*, pages 112–124. Springer, 2015.

[9] Piero A. Bonatti, Marco Faella, and Luigi Sauro. Defeasible Inclusions in Low-Complexity DLs. *Artificial Intelligence (JAIR)*, 42:719–764, 2011.

[10] Piero A. Bonatti, Carsten Lutz, and Frank Wolter. Expressive Non-Monotonic Description Logics Based on Circumscription. In Patrick Doherty, John Mylopoulos, and Christopher Welty, editors, *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR'06)*, pages 400–410. AAAI Press, 2009.

[11] Piero A. Bonatti, Carsten Lutz, and Frank Wolter. The Complexity of Circumscription in Description Logic. *Journal of Artificial Intelligence Research*, 35:717–773, 2009.

[12] Alexander Borgida and Luciano Serafini. Distributed Description Logics: Assimilating Information from Peer Sources. *Journal of Data Semantics*, 1:153–184, 2003.

[13] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. C-OWL: Contextualizing Ontologies. In Dieter Fensel, Katia P. Sycara, and John Mylopoulos, editors, *The Semantic Web - ISWC 2003, Second International Semantic Web Conference, Sanibel Island, FL, USA, October 20-23, 2003, Proceedings*, volume 2870 of *Lecture Notes in Computer Science*, pages 164–179. Springer, 2003.

[14] Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable Reasoning in Terminological Knowledge Representation Systems. In Ruzena Bajcsy, editor, *Proceedings of the 13th International Joint Conference on Artifical intelligence, IJCAI'93 – Volume 1*, pages 704–709, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[15] D. Carral Martínez, A. Krisnadhi, F. Maier, K. Sengupta, and P. Hitzler. Reconciling OWL and Rules. Technical report, DaseLab, Wright State University, Dayton, Ohio, USA, 2011. Available from http://www.pascal-hitzler.de/.

[16] Giovanni Casini, Thomas Meyer, Kodylan Moodley, and Riku Nortje. Relevant Closure: A New Form of Defeasible Reasoning for Description Logics. In Eduardo Fermé and João Leite, editors, *Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings*, volume 8761, pages 92–106. Springer, 2014.

[17] Giovanni Casini, Thomas Meyer, Kodylan Moodley, and Ivan José Varzinczak. Towards Practical Defeasible Reasoning for Description Logics. In Thomas Eiter, Birte Glimm, Yevgeny Kazakov, and Markus Krötzsch, editors, *Informal Proceedings of*

the 26th International Workshop on Description Logics, Ulm, Germany, July 23 - 26, 2013, volume 1014, pages 587–599. CEUR-WS.org, 2013.

[18] Giovanni Casini, Thomas Meyer, Ivan José Varzinczak, and Kodylan Moodley. Nonmonotonic Reasoning in Description Logics: Rational Closure for the ABox. In Thomas Eiter, Birte Glimm, Yevgeny Kazakov, and Markus Krötzsch, editors, *Informal Proceedings of the 26th International Workshop on Description Logics, Ulm, Germany, July 23 - 26, 2013*, volume 1014 of *CEUR Workshop Proceedings*, pages 600–615. CEUR-WS.org, 2013.

[19] Michelle Cheatham and Pascal Hitzler. String Similarity Metrics for Ontology Alignment. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul T. Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha F. Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web – ISWC 2013 – 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II*, volume 8219 of *Lecture Notes in Computer Science*, pages 294–309. Springer, 2013.

[20] Minh Dao-Tran, Thomas Eiter, and Thomas Krennwallner. Realizing Default Logic over Description Logic Knowledge Bases. In Claudio Sossai and Gaetano Chemello, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 10th European Conference, ECSQARU 2009, Verona, Italy, July 1-3, 2009. Proceedings*, volume 5590 of *Lecture Notes in Computer Science*, pages 602–613. Springer, 2009.

[21] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Andrea Schaerf, and Werner Nutt. An Epistemic Operator for Description Logics. *Artificial Intelligence*, 100(1-2):225–274, 1998.

[22] Francesco M. Donini, Daniele Nardi, and Riccardo Rosati. Autoepistemic Description Logics. In Martha E. Pollack, editor, *Proceedings of the Fifteenth International*

Joint Conference on Artificial Intelligence, IJCAI 97, Nagoya, Japan, August 23-29, 1997, 2 Volumes, pages 136–141. Morgan Kaufmann, 1997.

[23] Francesco M. Donini, Daniele Nardi, and Riccardo Rosati. Description Logics of Minimal Knowledge and Negation as Failure. *ACM Transactions on Computational Logic (TOCL)*, 3(2):177–225, April 2002.

[24] Zlatan Dragisic, Kai Eckert, Jérôme Euzenat, Daniel Faria, Alfio Ferrara, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Andreas Oskar Kempf, Patrick Lambrix, Stefano Montanelli, Heiko Paulheim, Dominique Ritze, Pavel Shvaiko, Alessandro Solimando, Cássia Trojahn dos Santos, Ondrej Zamazal, and Bernardo Cuenca Grau. Results of the Ontology Alignment Evaluation Initiative 2014. In Pavel Shvaiko, Jérôme Euzenat, Ming Mao, Ernesto Jiménez-Ruiz, Juanzi Li, and Axel Ngonga, editors, *Proceedings of the 9th International Workshop on Ontology Matching collocated with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Trentino, Italy, October 20, 2014.*, pages 61–104, 2014.

[25] Thomas Eiter, Giovambattista Ianni, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits. Combining Answer Set Programming with Description logics for the Semantic Web. *Artificial Intelligence*, 172(12-13):1495–1539, August 2008.

[26] Thomas Eiter, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits. Well-Founded Semantics for Description Logic Programs in the Semantic Web. In Grigoris Antoniou and Harold Boley, editors, *Rules and Rule Markup Languages for the Semantic Web: 3rd International Workshop, RuleML 2004*, volume 3323 of *Lecture Notes in Computer Science*, pages 81–97. Springer, 2004.

[27] Jérôme Euzenat, Christian Meilicke, Heiner Stuckenschmidt, Pavel Shvaiko, and Cássia Trojahn dos Santos. Ontology Alignment Evaluation Initiative: Six Years of Experience. *Journal Data Semantics*, 15:158–192, 2011.

[28] Achille Fokoue, Aaron Kershenbaum, Li Ma, Edith Schonberg, and Kavitha Srinivas. The Summary Abox: Cutting Ontologies Down to Size. In Isabel F. Cruz, Stefan Decker, Dean Allemang, Chris Preist, Daniel Schwabe, Peter Mika, Michael Uschold, and Lora Aroyo, editors, *The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings*, volume 4273 of *Lecture Notes in Computer Science*, pages 343–356. Springer, 2006.

[29] Michael Gelfond and Vladimir Lifschitz. The Stable Model Semantics for Logic Programming. In Robert A. Kowalski and Kenneth A. Bowen, editors, *Logic Programming. Proceedings of the 5th International Conference and Symposium on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[30] Michael Gelfond and Vladimir Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9(3/4):365–386, 1991.

[31] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. A Nonmonotonic Description Logic for Reasoning About Typicality. *Artificial Intelligence*, 195:165–202, 2013.

[32] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: An OWL 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[33] Ana Gomes, Jos Alferes, and Terrance Swift. Implementing Query Answering for Hybrid MKNF Knowledge Bases. In Manuel Carro and Ricardo Peña, editors, *Practical Aspects of Declarative Languages*, volume 5937 of *Lecture Notes in Computer Science*, pages 25–39. Springer Berlin / Heidelberg, 2010.

[34] Georg Gottlob. Complexity Results for Nonmonotonic Logics. *Journal of Logic and Computation*, 2(3):397–425, 1992.

[35] Stephan Grimm and Pascal Hitzler. Semantic Matchmaking of Web Resources with Local Closed-World Reasoning. *International Journal of Electronic Commerce*, 12(2):89–126, 2007.

[36] Stephan Grimm and Pascal Hitzler. A Preferential Tableaux Calculus for Circumscriptive $\mathcal{ALCO}$. In Axel Polleres and Terrance Swift, editors, *Proc. of the 3rd Int. Conference on Web Reasoning and Rule Systems (RR'09)*, volume 5837 of *Lecture Notes in Computer Science*, pages 40–54. Springer Berlin, 2009.

[37] Harry Halpin, Patrick J. Hayes, James P. McCusker, Deborah L. McGuinness, and Henry S. Thompson. When owl:sameAs Isn't the Same: An Analysis of Identity in Linked Data. In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks, and Birte Glimm, editors, *The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part I*, volume 6496 of *Lecture Notes in Computer Science*, pages 305–320. Springer, 2010.

[38] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph, editors. *OWL 2 Web Ontology Language: Primer*. W3C Recommendation 27 October 2009, 2009. Available from http://www.w3.org/TR/owl2-primer/.

[39] Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC, 2009.

[40] Pascal Hitzler and Anthony K. Seda. *Mathematical Aspects of Logic Programming Semantics*. CRC Press, 2010.

[41] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2(1):11–21, 2011.

[42] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible SROIQ. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors, *Proceedings,*

Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom, June 2-5, 2006, pages 57–67. AAAI Press, 2006.

[43] Ian Horrocks and Ulrike Sattler. Ontology Reasoning in the SHOQ(D) Description Logic. In Bernhard Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*, pages 199–204. Morgan Kaufmann, 2001.

[44] Shasha Huang and Qingguo Li. Reasoning with Vagueness in Hybrid MKNF Knowledge Bases. *Journal of Intelligent and Fuzzy Systems*, 26(4):1759–1770, 2014.

[45] Prateek Jain, Pascal Hitzler, Peter Z. Yeh, Kunal Verma, and Amit P. Sheth. Linked Data is Merely More Data. In Dan Brickley, Vinay K. Chaudhri, Harry Halpin, and Deborah McGuinness, editors, *Linked Data Meets Artificial Intelligence*, pages 82–86. AAAI Press, Menlo Park, CA, 2010.

[46] Krzysztof Janowicz. The Role of Space and Time for Knowledge Organization on the Semantic Web. *Semantic Web*, 1(1–2):25–32, 2010.

[47] Krzysztof Janowicz and Pascal Hitzler. The Digital Earth as Knowledge Engine. *Semantic Web*, 3(3):213–221, 2012.

[48] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. LogMap: Logic-Based and Scalable Ontology Matching. In Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy, and Eva Blomqvist, editors, *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, volume 7031 of *Lecture Notes in Computer Science*, pages 273–288. Springer, 2011.

[49] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Ontology Integration Using Mappings: Towards Getting the Right Logical

Consequences. In Lora Aroyo, Paolo Traverso, Fabio Ciravegna, Philipp Cimiano, Tom Heath, Eero Hyvönen, Riichiro Mizoguchi, Eyal Oren, Marta Sabou, and Elena Paslaru Bontas Simperl, editors, *The Semantic Web: Research and Applications, 6th European Semantic Web Conference, ESWC 2009, Heraklion, Crete, Greece, May 31-June 4, 2009, Proceedings*, volume 5554 of *Lecture Notes in Computer Science*, pages 173–187. Springer, 2009.

[50] Ernesto Jiménez-Ruiz, Christian Meilicke, Bernardo Cuenca Grau, and Ian Horrocks. Evaluating Mapping Repair Systems with Large Biomedical Ontologies. In Thomas Eiter, Birte Glimm, Yevgeny Kazakov, and Markus Krötzsch, editors, *Informal Proceedings of the 26th International Workshop on Description Logics, Ulm, Germany, July 23 - 26, 2013*, volume 1014 of *CEUR Workshop Proceedings*, pages 246–257, 2013.

[51] Amit Krishna Joshi, Prateek Jain, Pascal Hitzler, Peter Z. Yeh, Kunal Verma, Amit P. Sheth, and Mariana Damova. Alignment-Based Querying of Linked Open Data. In Robert Meersman, Hervé Panetto, Tharam S. Dillon, Stefanie Rinderle-Ma, Peter Dadam, Xiaofang Zhou, Siani Pearson, Alois Ferscha, Sonia Bergamaschi, and Isabel F. Cruz, editors, *On the Move to Meaningful Internet Systems: OTM 2012, Confederated International Conferences: CoopIS, DOA-SVI, and ODBASE 2012, Rome, Italy, September 10-14, 2012. Proceedings, Part II*, volume 7566 of *Lecture Notes in Computer Science*, pages 807–824. Springer, 2012.

[52] Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. The Incredible ELK - From Polynomial Procedures to Efficient Reasoning with $\mathcal{EL}$ Ontologies. *Journal of Automated Reasoning*, 53(1):1–61, 2014.

[53] Johan De Kleer and Kurt Konolige. Eliminating the Fixed Predicates from a Circumscription. *Artificial Intelligence*, 39(3):391–398, 1989.

[54] Matthias Knorr, José Júlio Alferes, and Pascal Hitzler. A Coherent Well-founded Model for Hybrid MKNF Knowledge Bases. In Malik Ghallab, Constantine D. Spyropoulos, Nikos Fakotakis, and Nikolaos M. Avouris, editors, *ECAI 2008 - 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25, 2008, Proceedings*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 99–103. IOS Press, 2008.

[55] Matthias Knorr, José Júlio Alferes, and Pascal Hitzler. Local Closed World Reasoning with Description Logics Under the Well-Founded Semantics. *Artificial Intelligence*, 175(9-10):1528–1554, 2011.

[56] Adila Krisnadhi, Frederick Maier, and Pascal Hitzler. OWL and Rules. In Axel Polleres, Claudia d'Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter F. Patel-Schneider, editors, *Reasoning Web. Semantic Technologies for the Web of Data - 7th International Summer School 2011, Galway, Ireland, August 23-27, 2011, Tutorial Lectures*, volume 6848 of *Lecture Notes in Computer Science*, pages 382–415. Springer, Heidelberg, 2011.

[57] Adila Krisnadhi, Kunal Sengupta, and Pascal Hitzler. Local Closed World Semantics: Keep it Simple, Stupid! In Riccardo Rosati, Sebastian Rudolph, and Michael Zakharyaschev, editors, *2011 International Workshop on Description Logics*, volume 745 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

[58] Markus Krötzsch, Frederick Maier, Adila A. Krisnadhi, and Pascal Hitzler. A Better Uncle for OWL: Nominal Schemas for Integrating Rules and Ontologies. In Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar, editors, *Proceedings of the 20th International World Wide Web Conference, WWW2011, Hyderabad, India, 2011*, pages 645–654. ACM, New York, 2011.

[59] Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler. ELP: Tractable Rules for OWL 2. In Amit P. Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy W. Finin, and Krishnaprasad Thirunarayan, editors, *Proceedings of the 7th International Semantic Web Conference (ISWC-08)*, volume 5318 of *Lecture Notes in Computer Science*, pages 649–664. Springer, 2008.

[60] Patrick Lambrix, Fang Wei-Kleiner, Zlatan Dragisic, and Valentina Ivanova. Repairing Missing is-a Structure in Ontologies is an Abductive Reasoning Problem. In Patrick Lambrix, Guilin Qi, Matthew Horridge, and Bijan Parsia, editors, *Proceedings of the Second International Workshop on Debugging Ontologies and Ontology Mappings, Montpellier, France, May 27, 2013*, volume 999, pages 33–44. CEUR-WS.org, 2013.

[61] Nicola Leone and Wolfgang Faber. The DLV Project: A Tour from Theory and Research to Applications and Market. In Maria Garcia de la Banda and Enrico Pontelli, editors, *Logic Programming, 24th International Conference, ICLP 2008, Udine, Italy, December 9-13 2008, Proceedings*, volume 5366 of *Lecture Notes in Computer Science*, pages 53–68. Springer, 2008.

[62] Yue Ma and Pascal Hitzler. Paraconsistent Reasoning for OWL 2. In Axel Polleres and Terrance Swift, editors, *Web Reasoning and Rule Systems, Third International Conference, RR 2009, Chantilly, VA, USA, October 25-26, 2009, Proceedings*, volume 5837 of *Lecture Notes in Computer Science*, pages 197–211. Springer, 2009.

[63] Despoina Magka, Markus Krötzsch, and Ian Horrocks. Computing Stable Models for Nonmonotonic Existential Rules. In Francesca Rossi, editor, *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*. IJCAI/AAAI, 2013.

[64] Frederick Maier. Extending Paraconsistent $\mathcal{SROIQ}$. In Pascal Hitzler and Thomas Lukasiewicz, editors, *Web Reasoning and Rule Systems - Fourth International Conference, RR 2010, Bressanone/Brixen, Italy, September 22-24, 2010. Proceedings*, volume 6333 of *Lecture Notes in Computer Science*, pages 118–132. Springer, 2010.

[65] Frederick Maier, Yue Ma, and Pascal Hitzler. Paraconsistent OWL and Related Logics. *Semantic Web*, 4(4):395–427, 2013.

[66] David Makinson. *Bridges from Classical to Nonomonotonic Logic*, volume 5 of *Texts in Computing*. King's College Publications, 2005.

[67] David Carral Martínez, Krzysztof Janowicz, and Pascal Hitzler. A Logical Geo-Ontology Design Pattern for Quantifying over Types. In Isabel F. Cruz, Craig A. Knoblock, Peer Kröger, Egemen Tanin, and Peter Widmayer, editors, *SIGSPATIAL 2012 International Conference on Advances in Geographic Information Systems (formerly known as GIS), SIGSPATIAL'12, Redondo Beach, CA, USA, November 7-9, 2012*, pages 239–248. ACM, 2012.

[68] Brian McBride. The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 51–66. Springer, 2004.

[69] John McCarthy. Circumscription – A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, 13(1–2):27–39, 1980.

[70] Christian Meilicke. The Relevance of Reasoning and Alignment Incoherence in Ontology Matching. In Lora Aroyo, Paolo Traverso, Fabio Ciravegna, Philipp Cimiano, Tom Heath, Eero Hyvönen, Riichiro Mizoguchi, Eyal Oren, Marta Sabou, and Elena Paslaru Bontas Simperl, editors, *The Semantic Web: Research and Applications, 6th European Semantic Web Conference, ESWC 2009, Heraklion, Crete, Greece, May*

31-June 4, 2009, Proceedings, volume 5554 of *Lecture Notes in Computer Science*, pages 934–938. Springer, 2009.

[71] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Repairing Ontology Mappings. In Robert C. Holte and Adele Howe, editors, *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 1408–1413, 2007.

[72] Robert Moore. Possible-worlds Semantics for Autoepistemic Logic. In *Proceedings of the 1984 Non-monotonic Reasoning Workshop*. AAAI, Menlo Park, CA, 1984.

[73] Robert Moore. Semantical Considerations on Nonmonotonic Logic. *Artificial Intelligence*, 25(1), 1985.

[74] Boris Motik, Ian Horrocks, and Ulrike Sattler. Adding Integrity Constraints to OWL. In Christine Golbreich, Aditya Kalyanpur, and Bijan Parsia, editors, *Proceedings of the OWLED 2007 Workshop on OWL: Experiences and Directions*, volume 258, 2007.

[75] Boris Motik and Riccardo Rosati. Reconciling Description Logics and Rules. *Journal of the ACM*, 57(5):1–62, 2010.

[76] Boris Motik, Ulrike Sattler, and Rudi Studer. Query Answering for OWL-DL with Rules. *Journal of Web Semantics*, 3:41–60, July 2005.

[77] Daniel Oberle, Anupriya Ankolekar, Pascal Hitzler, Philipp Cimiano, Michael Sintek, Malte Kiesel, Babak Mougouie, Stephan Baumann, Shankar Vembu, and Massimo Romanelli. DOLCE ergo SUMO: On Foundational and Domain Models in the SmartWeb Integrated Ontology (SWIntO). *J. Web Sem.*, 5(3):156–174, 2007.

[78] Dantong Ouyang, Xianji Cui, and Yuxin Ye. Integrity Constraints in OWL Ontologies Based on Grounded Circumscription. *Frontiers of Computer Science*, 7(6):812–821, 2013.

[79] Chintan Patel, James J. Cimino, Julian Dolby, Achille Fokoue, Aditya Kalyanpur, Aaron Kershenbaum, Li Ma, Edith Schonberg, and Kavitha Srinivas. Matching Patient Records to Clinical Trials Using Ontologies. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 816–829. Springer, 2007.

[80] Raymond Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13(1-2):81–132, 1980.

[81] Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. Cheap Boolean Role Constructors for Description Logics. In Steffen Hölldobler, Carsten Lutz, and Heinrich Wansing, editors, *Logics in Artificial Intelligence, 11th European Conference, JELIA 2008, Dresden, Germany, September 28 - October 1, 2008. Proceedings*, volume 5293, pages 362–374. Springer, 2008.

[82] Kunal Sengupta and Pascal Hitzler. Web Ontology Language (OWL). In *Encyclopedia of Social Network Analysis and Mining*, pages 2374–2378. 2014.

[83] Kunal Sengupta, Pascal Hitzler, and Krzysztof Janowicz. Revisiting Default Description Logics – and Their Role in Aligning Ontologies. In Thepchai Supnithi, Takahira Yamaguchi, Jeff Z. Pan, Vilas Wuwongse, and Marut Buranarach, editors, *Semantic Technology, 4th Joint International Conference, JIST 2014, Chiang Mai, Thailand, November 9-11, 2014*, volume 8943, pages 3–18. Springer, 2015.

[84] Kunal Sengupta, Adila Alfa Krisnadhi, and Pascal Hitzler. Local Closed World Semantics: Grounded Circumscription for OWL. In Lora Aroyo, Chris Welty, Harith

Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy, and Eva Blomqvist, editors, *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, volume 7031 of *Lecture Notes in Computer Science*, pages 617–632. Springer, 2011.

[85] Luciano Serafini, Alexander Borgida, and Andrei Tamilin. Aspects of Distributed and Modular Ontology Reasoning. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005*, pages 570–575. Professional Book Center, 2005.

[86] Luciano Serafini and Andrei Tamilin. Local Tableaux for Reasoning in Distributed Description Logics. In Volker Haarslev and Ralf Möller, editors, *Proceedings of the 2004 International Workshop on Description Logics (DL2004), Whistler, British Columbia, Canada, June 6-8, 2004*, volume 104 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.

[87] Inanç Seylan, Enrico Franconi, and Jos de Bruijn. Effective Query Rewriting with Ontologies over DBoxes. In Craig Boutilier, editor, *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pages 923–925, 2009.

[88] Pavel Shvaiko and Jérôme Euzenat. Ontology Matching: State of the Art and Future Challenges. *IEEE Transanctions in Knowledge and Data Engineering*, 25(1):158–176, 2013.

[89] Heiner Stuckenschmidt, Frank van Harmelen, Paolo Bouquet, Fausto Giunchiglia, and Luciano Serafini. Using C-OWL for the Alignment and Merging of Medical Ontologies. In Udo Hahn, editor, *KR-MED 2004, First International Workshop on Formal Biomedical Knowledge Representation, Proceedings of the KR 2004 Workshop on*

Formal Biomedical Knowledge Representation, Whistler, BC, Canada, 1 June 2004, volume 102 of *CEUR Workshop Proceedings*, pages 88–101. CEUR-WS.org, 2004.

[90] Jiao Tao, Evren Sirin, Jie Bao, and Deborah L. McGuinness. Integrity Constraints in OWL. In Maria Fox and David Poole, editors, *Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI 2010*. AAAI Press, 2010.

[91] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD thesis, RWTH Aachen, Germany, 2001.

[92] Markel Vigo, Caroline Jay, and Robert Stevens. Protégé4US: Harvesting Ontology Authoring Data with Protégé. In Valentina Presutti, Eva Blomqvist, Raphaël Troncy, Harald Sack, Ioannis Papadakis, and Anna Tordai, editors, *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, volume 8798 of *Lecture Notes in Computer Science*, pages 86–99. Springer, 2014.

A Appendix

Ontologies for Experiment on Marriage Example

Marriage Ontology 1 in OWL Functional Syntax

Prefix(:=<http://daselab.org/ontologies/marriage2#>)

Prefix(owl:=<http://www.w3.org/2002/07/owl#>)

Prefix(rdf:=<http://www.w3.org/1999/02/22-rdf-syntax-ns#>)

Prefix(xml:=<http://www.w3.org/XML/1998/namespace>)

Prefix(xsd:=<http://www.w3.org/2001/XMLSchema#>)

Prefix(rdfs:=<http://www.w3.org/2000/01/rdf-schema#>)

Prefix(marriage2:=<http://daselab.org/example/marriage2#>)

Ontology(<http://daselab.org/example/marriage2>

Declaration(Class(marriage2:Female))

Declaration(Class(marriage2:Male))

Declaration(ObjectProperty(marriage2:hasSpouse))

Declaration(NamedIndividual(marriage2:david))

ClassAssertion(marriage2:Male marriage2:david)

ObjectPropertyAssertion(marriage2:hasSpouse marriage2:david marriage2:mike)

Declaration(NamedIndividual(marriage2:jacob))

ClassAssertion(marriage2:Male marriage2:jacob)

ObjectPropertyAssertion(marriage2:hasSpouse marriage2:jacob marriage2:jane)

Declaration(NamedIndividual(marriage2:jane))

ClassAssertion(marriage2:Female marriage2:jane)

Declaration(NamedIndividual(marriage2:julie))

ClassAssertion(marriage2:Female marriage2:julie)

Declaration(NamedIndividual(marriage2:mark))

ClassAssertion(marriage2:Male marriage2:mark)

ObjectPropertyAssertion(marriage2:hasSpouse marriage2:mark marriage2:julie)

Declaration(NamedIndividual(marriage2:mike))

ClassAssertion(marriage2:Male marriage2:mike)

)

Marriage Ontology 2 in OWL Functional Syntax

Prefix(:=<http://www.semanticweb.org/kunal/ontologies/2015/6/marriage1#>)

Prefix(owl:=<http://www.w3.org/2002/07/owl#>)

Prefix(rdf:=<http://www.w3.org/1999/02/22-rdf-syntax-ns#>)

Prefix(xml:=<http://www.w3.org/XML/1998/namespace>)

Prefix(xsd:=<http://www.w3.org/2001/XMLSchema#>)

Prefix(rdfs:=<http://www.w3.org/2000/01/rdf-schema#>)

Ontology(<http://www.daselab.org/example/marriage1>

Declaration(Class(:Female))

DisjointClasses(:Female :Male)

Declaration(Class(:FemaleSpouse))

EquivalentClasses(:FemaleSpouse ObjectSomeValuesFrom(:hasSpouse :Female))

SubClassOf(:FemaleSpouse :Male)

Declaration(Class(:Male))

DisjointClasses(:Male :Female)

Declaration(Class(:MaleSpouse))

EquivalentClasses(:MaleSpouse ObjectSomeValuesFrom(:hasSpouse :Male))

SubClassOf(:MaleSpouse :Female)

Declaration(ObjectProperty(:hasSpouse))

Declaration(NamedIndividual(:john))

ClassAssertion(:Male :john)

ObjectPropertyAssertion(:hasSpouse :john :mary)

Declaration(NamedIndividual(:mary))

ClassAssertion(:Female :mary)

)

Output of Experiment 2 - Biomedical Ontologies

Printing all instances of class Space_of_compartment_of_trunk>

<ind1101>

<ind1100>

Printing all instances of class Parenchymatous_organ>

<ind1000>

Printing all instances of class Cardinal_organ_part>

<ind1103>

<ind1102>

Printing all instances of class Material_anatomical_entity>

<ind1102>

<ind1000>

<ind1103>

Printing all instances of class Membrane>

<ind1103>

<ind1102>

Printing all instances of class Anatomical_structure>

<ind1000>

<ind1102>

<ind1103>

Printing all instances of class Physical_anatomical_entity>

<ind1101>

<ind1100>

<ind1102>

<ind1000>

<ind1103>

Printing all instances of class Anatomical_entity_template>

<ind1100>

<ind1102>

<ind1000>

<ind1101>

<ind1103>

Printing all instances of class General_anatomical_term>

<ind1103>

<ind1102>

Printing all instances of class Anatomical_compartment_space>

<ind1101>

<ind1100>

Printing all instances of class Attribute_entity>

<ind1102>

<ind1103>

Printing all instances of class Set_of_organs>

<ind1000>

Printing all instances of class Set_of_viscera>

<ind1000>

Printing all instances of class Organ>

<ind1000>

Printing all instances of class Lobular_organ>

<ind1000>

Printing all instances of class Anatomical_entity>

<ind1103>

<ind1000>

<ind1101>

<ind1100>

<ind1102>

Printing all instances of class Anatomical_space>

<ind1101>

<ind1100>

Printing all instances of class Solid_organ>

<ind1000>

Printing all instances of class Lung>

<ind1000>

Printing all instances of class Thoracic_cavity>

<ind1101>

<ind1100>

Printing all instances of class Immaterial_anatomical_entity>

<ind1100>

<ind1101>

Printing all instances of class Standard_FMA_class>

<ind1000>

<ind1101>

<ind1100>

<ind1102>

<ind1103>

Printing all instances of class Anatomical_set>

<ind1000>

Printing all instances of class Miscellaneous_term>

<ind1103>

<ind1102>

Printing all instances of class Pair_of_lungs>

<ind1000>

Printing all instances of class Body_cavity_subdivision>

<ind1101>

<ind1100>