AVERTISSEMENT

LIENS

Code la Propriété Intellectuelle – Articles L. 122-4 et L. 335-1 à L. 335-10
Loi n°92-597 du 1er juillet 1992, publiée au *Journal Officiel* du 2 juillet 1992
http://www.cfcopies.com/V2/leg/leg-droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

THESIS

in partial fulfillment of the requirements for the degree of

PHD

Delivered by *the University of Toulouse*

Knowledge base ontological debugging guided by linguistic evidence

by **Julien CORMAN**

JURY

Laure VIEU	Senior researcher	Supervisor
Nathalie AUSSENAC-GILLES	Senior researcher	Co-supervisor
Renata WASSERMANN	Professor	Rapporteur
Bernardo MAGNINI	Senior researcher	Rapporteur
Marie-Christine ROUSSET	Professor	Examiner
Bernardo CUENCA-GRAU	Professor	Examiner
Fabian SUCHANEK	Associate professor	Examiner

Doctoral School and Specialty:
MITT: Domaine STIC: Intelligence Artificielle

Research Unit:
Institut de Recherche en Informatique de Toulouse (UMR 5505)

Supervisors:
Laure VIEU and *Nathalie AUSSENAC-GILLES*

Rapporteurs:
Renata WASSERMANN and *Bernardo MAGNINI*

Abstract

When they grow in size, knowledge bases (KBs) tend to include sets of axioms which are intuitively absurd but nonetheless logically consistent. This is particularly true of data expressed in OWL, as part of the Semantic Web framework, which favors the aggregation of set of statements from multiple sources of knowledge, with overlapping signatures.

Identifying nonsense is essential if one wants to avoid undesired inferences, but the sparse usage of negation within these datasets generally prevents the detection of such cases on a strict logical basis. And even if the KB is inconsistent, identifying the axioms responsible for the nonsense remains a non trivial task.

This thesis investigates the usage of automatically gathered linguistic evidence in order to detect and repair violations of common sense within such datasets. The main intuition consists in exploiting distributional similarity between named individuals of an input KB, in order to identify consequences which are unlikely to hold if the rest of the KB does. Then the repair phase consists in selecting axioms to be preferably discarded (or at least amended) in order to get rid of the nonsense.

A second strategy is also presented, which consists in strengthening the input KB with a foundational ontology, in order to obtain an inconsistency, before performing a form of knowledge base debugging/revision which incorporates this linguistic input.

This last step may also be applied directly to an inconsistent input KB.

These propositions are evaluated with different sets of statements issued from the Linked Open Data cloud, as well as datasets of a higher quality, but which were automatically degraded for the evaluation. The results seem to indicate that distributional evidence may actually constitute a relevant common ground for deciding between conflicting axioms.

Contents

List of Figures

List of Tables

Acknowledgements

First and foremost, I want to thank my supervisors, Laure Vieu and Nathalie Aussenac-Gilles, for their guidance, patience and (almost unconditional) support during these three years.

I would also like to thank the members of my jury, Renata Wassermann, Bernardo Magnini, Marie-Christine Rousset, Bernardo Cuenca-Grau and Fabian Suchanek, for the close attention they paid to our work, the time and efforts they spent reviewing it, and the useful insights they provided.

Thanks also once again to Renata Wassermann, as well as Giancarlo Guizzardi and the members of their respective research groups for their warm welcome during my stay in Brazil.

Finally, I want to thank my labmates (and friends) for their support and precious help.

I benefited from arguably ideal conditions to write this thesis, in a very stimulating environment, and am very grateful for that, not only to the people just mentioned, but also to all members of the IRIT or University of Toulouse I have had the occasion to work with but did not mention here.

Chapter 1

Introduction

A knowledge base (KB) may be described as an elaborated form of database, more expressive than a traditional relational database (RDB), with a focus on (mostly deductive) reasoning, in order for instance to infer additional knowledge out of the explicitly expressed one.

As a very simple illustration, just like a RDB, a KB may express the fact that "Reva Gerstein was born in Toronto". In first-order logic (FOL), this could be expressed with the formula:

(1) $\texttt{birthplace}(\textit{Reva Gerstein}, \textit{Toronto})$

This is actually a real statement, extracted from the KB DBPedia [MJB12],[1] as well as all examples used in this introduction, unless explicitly mentioned. The notation is simplified here for readability: each predicate (like `birthplace`) or individual constant (like *Reva Gerstein*) has a longer and non-ambiguous identifier in this KB (precisely, an IRI [2]), such that no confusion can be made with another *Reva Ger-*

[1] DBpedia will be introduced in more details in Chapter 5 Section 5.1.1

[2] For "Internationalized Resource Identifier". IRIs are extensions of URIs ("Universal Resource Identifiers") to Unicode characters.

stein, another *Toronto*, or another `birthplace`. Similarly, the following statement expresses the fact that the `occupation` of *Reva Gerstein* is *Psychologist*.

(2) `occupation`(*Reva Gerstein*, *Psychologist*)

A predicate may appear in multiple statements. For instance, `birthplace` and `occupation` also appear in the following ones:

(3) `birthplace`(*Rob Shuter*, *Birmingham*)

(4) `occupation`(*Rob Shuter*, *Gossip columnist*)

A KB may also express the fact that the second argument of the predicate `birthplace` is a place, i.e.:

(5) $\forall x, y : \texttt{birthplace}(x, y) \rightarrow \texttt{Place}(y)$

In this case, it can be inferred that *Toronto* and *Birmingham* are instances of the predicate `Place`, i.e. the formulas `Place`(*Toronto*) and `Place`(*Birmingham*) are consequences of statements 1 to 5, although they are not explicitly stated in the KB. Explicitly stated formulas like statements 1 to 5 are often called *axioms* in the knowledge representation community, and this is also the terminology adopted in this thesis. As a further illustration, here are a few additional axioms:[3]

(6) $\forall x : \texttt{Place}(x) \rightarrow \neg\texttt{Agent}(x)$

(7) $\forall x, y : \texttt{knownFor}(x, y) \rightarrow \texttt{Person}(x)$

(8) $\forall x : \texttt{Person}(x) \rightarrow \texttt{Agent}(x)$

Now let us assume that axioms 1 to 8 are extended with an additional axiom $\phi =$ `knownFor`(*Toronto*, *Canadian National Tower*).[4] Then the KB composed of these 9

[3] still extracted from DBpedia

[4] ϕ is not a real DBpedia statement, which is the only reason why it is not designated with axiom 9 here.

axioms is logically inconsistent: $\texttt{Agent}(\mathit{Toronto})$ is a consequence of axioms $\{7, 8\} \cup \{\phi\}$, but from axioms $\{1, 5, 6\}$, one may also infer $\neg\texttt{Agent}(\mathit{Toronto})$.

Important efforts have been devoted in the last 15 years to the promotion of such forms of knowledge representation (see [HKR09]). In particular, successful standards for data exchange have been designed, the most influential being arguably the ones developed by the W3C, as part of the Semantic Web (SW) project.

An important advantage of KBs and logic-based knowledge representation languages is flexibility. A KB can not only be viewed, but also manipulated as a set of logical statements: axioms can be added, deleted or edited, with very few constraints other than being syntactically valid. For instance, in the above example, the inconsistency can be solved by directly editing the axioms of the KB. Among other possibilities,[5] one may decide to get rid of axiom 7, 8 or ϕ, or even weaken axiom 5 into $\forall x, y : \texttt{birthplace}(x, y) \rightarrow (\texttt{Place}(y) \vee \texttt{Organization}(y))$. An important difference from RDBs is that both the data (sometimes called the *ABox*, i.e axioms $\{1, 2, 3, 4\} \cup \{\phi\}$ above) and the data schema (sometimes called the *TBox*, i.e. axioms {5,6,7,8}) of a KB may be edited. In contrast, the data schema of a RDB is viewed as constraining the data, and cannot be easily updated without compromising the data.

Another important advantage is modularity (see for instance [SSZ09] for an introduction). It is possible in theory to build a KB for a given application out of fragments of other KBs. For instance, in the above example, one may be interested in knowledge pertaining to the city of Toronto only, in which case only axioms {1,5,6,7,8} and ϕ are relevant (the case of (2) remains debatable). Other Toronto related axioms may be added from other source KBs, provided they are expressed

[5] Potentially an infinite number of them, if "solving the inconsistency" is not defined more formally. This notion will be made more precise in the following.

in the same logic (or a less expressive one), yielding a Toronto focused KB. Ideally, if the overlap in the signatures of the different source KBs goes beyond Toronto, i.e. if other predicates or individual constants are shared by these sources (e.g. `Agent`, `birthPlace` or *Reva Gerstein*), some additional knowledge may be inferred from the Toronto KB, which could not be inferred from the source KBs individually. This is one of the reasons why shared vocabularies among KBs (or alternatively *alignments*[6] between the signatures of different KBs, see [ESo07]) are at the core of the SW project.

Flexibility and modularity can both be viewed as a blessing and a curse though. When they grow in size or integrate axioms from multiple sources, KBs also tend to have undesired properties, one of the most obvious ones being logical inconsistency, illustrated above. But a consistent KB may have undesired properties as well. These may be computational: for instance, in some very expressive Description Logics (such as $\mathcal{SROIQ}$, described in [HKS06]), even if all axioms of a consistent KB are syntactically valid when considered individually, this may not be sufficient to ensure decidability of entailment, and additional syntactic constraints on the KB as a whole may need to be respected. Even in less expressive logics, some (different) constraints on the KB as a whole may be needed in order to ensure tractability of some reasoning tasks.

But a consistent KB may also have undesired semantic properties, which is the focus of the work presented in this thesis. As an illustration, consider the above

[6] A set A of alignments is a set of formulas expressing equivalence (or sometimes subsumption) between predicates or individual constants from the signatures of two KBs K_1 and K_2. For instance, an alignment may express the fact that the predicate `PhysicalPlace` in K_1 is equivalent to the predicate `Place` in K_2, or more specific. More complex alignments are also possible, involving multiple predicates or individual constants appearing in K_1 and K_2. But alignments are themselves axioms, so formally, one may as well consider that the signatures of $K_1 \cup A$ and K_2 overlap. Therefore in the following, a "signature overlap" will be used as a generic term to designate both genuine overlap and overlap with alignments.

example, without axiom ϕ, and extended with the two following DBpedia axioms:

(9) $\forall x, y : \texttt{occupation}(x, y) \rightarrow \texttt{PersonFunction}(y)$

(10) $\texttt{occupation}(\textit{Montgomery C. Meigs}, \textit{Smithsonian Institution})$

From axioms {2,4,9,10}, the *Smithsonian Institution* is a `PersonFunction` just like *Psychologist* and *Gossip columnist* are, which is arguably counterintuitive. But the KB composed of axioms {2,4,9,10} is nonetheless logically consistent. More importantly, it turns out that no information within DBpedia can be used to detect this nonsense on a logical basis (this claim will be made more explicit in section 1.1.2). Worse, such cases tend to be the norm rather than exceptions, because statements expressing some form of negation (like axiom 6) are very infrequent in practice, at least among datasets adopting the SW standards, as will be illustrated.

KB debugging deals with the detection and/or repair of undesired properties of an input KB. This is not restricted to inconsistency, and KB debugging encompasses a wide variety of techniques, reviewed in Chapter 3. For consistent KBs though, automated debugging remains limited to syntactic verifications which generally do not address the type of semantic errors just illustrated, or only accidentally. On the other hand, for inconsistent[7] KBs, current approaches suffer from the number of candidate outputs. Among other contributions, this thesis investigates the use of linguistic evidence automatically gathered from web pages in order to address both of these issues, which to our knowledge is an original proposal. Repair is understood as the automated weakening of the input KB. An additional phase may be needed afterwards, which consists in rephrasing the information lost during the weakening phase, but this posterior reformulation falls out of the scope of this work.

[7] or *incoherent* in its technical sense in the Description Logics community, defined in Chapter 2 Section 2.3.5

The focus is put on KBs expressed in OWL, a knowledge representation language based on Description Logics (DLs), and a widely used W3C standard. The latest version of OWL is OWL 2, defined by five W3C recommendations published in 2012.[8] DLs are decidable fragments of FOL, for which numerous algorithms have been designed, covering a variety of tasks, like checking satisfiability or subsumption, but also computing modules, minimal conflicts...[9]

Section 1.1 further illustrates and characterizes the problem at hand, whereas Section 1.2 lists some of the main constraints and commitments underlying this work, positioning it wrt to alternative existing approaches. Section 1.3 briefly introduces the concrete solutions proposed in this thesis, and Section 1.4 reviews the content of the different chapters.

1.1 Semantic Web and violations of common sense

1.1.1 Multiple meanings of a constant or predicate

OWL is a knowledge representation language designed as part of the SW framework, in order to publish, share and reuse structured data issued from multiple sources. In particular, data publishers are encouraged to reuse predicates or constants from already published datasets. In other words, interoperability between two OWL KBs is primarily understood as a signature overlap.

Integrating knowledge from different sources can easily lead to intuitively absurd sets of axioms though, due to multiple and incompatible meanings of the same constants or predicates. This may be the case when two different datasets with over-

[8] http://www.w3.org/TR/2012/REC-owl2-overview-20121211/#Documentation_Roadmap
[9] An introduction to Description Logics is provided in Chapter 2, Section 2.3.

lapping signature are (partially) merged, or queried altogether. But this phenomenon is also frequent within large datasets, in particular in cross-domain and (partially) collaborative KB such as Freebase [BEP+08] or DBpedia.

The introductory example given above remained very simple on purpose, and could easily be solved manually. The following is a more in-depth illustration of the issue, even though it is composed of 12 DBpedia axioms only. When the syntactic formulations differ, the axioms are given both in DL and FOL:

Ex 1.1.1. (1) $\texttt{keyPerson}(\textit{Caixa Bank}, \textit{CEO})$

(2) $\texttt{keyPerson}(\textit{Sina Bank}, \textit{CEO})$

(3) $\texttt{occupation}(\textit{Peter Munk}, \textit{CEO})$

(4) $\texttt{keyPerson}(\textit{BrookField Office Properties}, \textit{Peter Munk})$

(5) $\top \sqsubseteq \forall \texttt{keyPerson}.\texttt{Person}$

$\forall x, y : \texttt{keyPerson}(x, y) \rightarrow \texttt{Person}(y)$

(6) $\texttt{occupation}(\textit{Montgomery C. Meigs}, \textit{Smithsonian Institution})$

(7) $\top \sqsubseteq \forall \texttt{occupation}.\texttt{PersonFunction}$

$\forall x, y : \texttt{occupation}(x, y) \rightarrow \texttt{PersonFunction}(y)$

(8) $\exists \texttt{hasPersonName}.\top \sqsubseteq \texttt{PersonFunction}$

$\forall x, y : \texttt{hasPersonName}(x, y) \rightarrow \texttt{PersonFunction}(x)$

(9) $\texttt{occupation}(\textit{Ernest Noel}, \textit{Ernest Noel 1})$

(10) $\texttt{occupation}(\textit{Ernest Noel}, \textit{Ernest Noel 2})$

(11) $\texttt{employer}(\textit{Frederick Knab}, \textit{Smithsonian Institution})$

(12) $\top \sqsubseteq \forall \texttt{employer}.\texttt{Organisation}$

$\forall x, y : \texttt{employer}(x, y) \rightarrow \texttt{Organisation}(y)$

Axioms 1 to 5, when considered altogether, violate some common sense intuitions, for instance the fact that nothing (*CEO* here) should be both a key person

of something, and the occupation of someone (*Peter Munk*) who is himself a key person of something. Two intuitively incompatible meanings of `keyPerson` coexist. In axioms 1 and 2, the intending meaning of `keyPerson` seems to be "has as a key person someone whose occupation is", whereas it seems to be "has as a key person" in axioms 4 and 5.

Three different meanings of the predicate `occupation` can also be observed in axioms 3, 6, 7, 9 and 10: it takes an organization (*Smithsonian Institution*) as object in axiom 6, but a profession (*CEO*) in axiom 3. The meaning of `occupation` in axioms 9 and 10 is less obvious: *Ernest Noel 1* and *Ernest Noel 2* actually stand for "Ernest Noel as a businessman" and "Ernest Noel as a member of Parliament" respectively,[10] such that `occupation` ranges this time over individual roles (akin to the qua-individuals of [MGV+05]) which are specific to a given human being. This last view is reinforced by axiom 8, which suggests a mapping from such roles to person names. Finally, the intended meaning of `occupation` in axiom 7 remains unclear. It may be in line with its use in axiom 3, in which case a `PersonFunction` in axiom 7 should be understood as an activity (like *CEO*). Or `occupation` in axiom 7 should be understood as it is in axioms 9 and 10, in which case `PersonFunction` has the same meaning of individual role in axioms 7 and 8.

This kind of nonsense hinders the reliability of a KB, limiting its potential use. It may in particular lead to undesired inferences. For instance, according to axioms 1, 2 and 5, *Caixa Bank* and *Sina Bank* have the same `Person` as a `keyPerson`, which was certainly not intended, and turns out to be false (they do not share a key employee). This thesis investigates several strategies in order to automatically detect such violations of common sense, and suggest one or several weaker version(s)

[10] which can be seen from additional DBpedia statements, not reproduced here. This example is discussed in more details in Chapter 7 Section 7.1.

of the input KB as a repair.

Two additional useful observations can be made about this example:

- These axioms when taken individually do not convey erroneous information (like `director`(*Citizen Kane*, *Woody Allen*)), neither outdated information (like `president`(*USA*, *George W. Bush*)), but they are nonetheless absurd when considered together. Even axiom 6 (`occupation`(*Montgomery C. Meigs*, *Smithsonian Institution*)) has an intuitively clear (and factually correct) intended meaning, i.e. this use of `occupation` may appear as incorrect only if axiom 6 if compared to other axioms where `occupation` appears.

- Identifying mutually exclusive sets of axioms in this small sample is already non-trivial, which suggests that a manual repair for realistically sized KBs can be potentially very complex.

- In most cases, deciding from this sample only which of the two or three incompatible meanings of a same predicate is the correct one may arguably be perceived as an arbitrary choice. Instead, the dominant meaning of this predicate within the whole KB should probably be taken into account. For instance, if `keyPerson` is mostly understood within DBpedia as ranging over person functions like *CEO*, then statements 4 and 5 are the erroneous ones. If it mostly ranges over human beings instead (like *Peter Munk*), then statements 1 and 2 are the outliers.

1.1.2 Lack of negation and consistency by default

Another interesting observation about example 1.1.1 is the absence of explicit contradiction, i.e. the fact that this set of axioms is logically consistent,[11] even though intuitively absurd.

Some form of negation is required for an inconsistency to be derived. For instance, nothing in example 1.1.1 explicitly prevents a same individual to be both a `Person` and a `PersonFunction` (like *CEO* is), or both a `PersonFunction` and an `Organisation` (like the *Smithsonian Institution* is). It turns out that nothing prevents it in DBpedia either. More generally, aside from cardinality restrictions (like functional restriction on datatyProperties, described in Chapter 2 Section 2.3), negation is very sparsely used in datasets published as part of the SW project. As an illustration, according to the LODstats survey tool [ADML12], which provides statistics about a sample of the Linked Open Data Cloud (described in [BHBL09]), the two OWL constructs involving concept negation, namely `owl:disjointWith` and `owl:complementOf`, which are possibly the two most straightforward ways to express negation in OWL,[12] have been observed 333 times and twice respectively, against more that 89 000 occurrences of the construct `rdfs:subClassOf`.

This does not mean that these datasets are necessarily consistent.[13] For instance, taken as a whole, DBpedia is inconsistent, as illustrated by the following set of statements:

(a) `creator`(*News Bites*, *Studio 23*)

[11] and logically *coherent* in the DL sense, defined in Chapter 2 Section 2.3.5

[12] but not the only ones. Among other possibilities are the cardinality restrictions already mentioned, but also non-identity between individual constants (the unique name assumption is not made by default in OWL), and nominals. These constructs are reviewed in Chapter 2 Section 2.3.

[13] or coherent

(b) $\texttt{Place}(\textit{Studio 23})$

(c) $\top \sqsubseteq \forall \texttt{creator}.\texttt{Person}$

$\forall x, y : \texttt{creator}(x, y) \rightarrow \texttt{Person}(y)$

(d) $\texttt{Person} \sqsubseteq \texttt{Agent}$

$\forall x : \texttt{Person}(x) \rightarrow \texttt{Agent}(x)$

(e) $\texttt{Place} \sqsubseteq \neg\texttt{Agent}$

$\forall x : \texttt{Place}(x) \rightarrow \neg\texttt{Agent}(x)$

But the inconsistency of DBpedia is independent from the common sense violations identified among some of the 12 axioms of example 1.1.1. For instance, axioms 1 and 4 of example 1.1.1 involves two intuitively incompatible meanings of the predicate `keyPerson`. But there is no subset Q of DBpedia such that $\{1, 4\} \subseteq Q$, $Q \vdash \bot$, and for all $Q' \subset Q$, $Q' \nvdash \bot$. In other words, in any proof of the inconsistency of DBpedia, at least one of axioms 1 and 4 is not needed. This does not hold of {1,4} only, but also of all minimal intuitively absurd sets of axioms of example 1.1.1, like $\{2, 4\}, \{1, 5\}, \{3, 6\}, \{6, 7\}$, etc. As a consequence, even if it could be done, solving the inconsistency is very unlikely to address the problems identified in example 1.1.1 (at best marginally and as a non-intended side effect, provided axiom 1 or axiom 4 for instance appears in a minimal inconsistent subset Q of DBpedia).

Another consequence of this sparse usage of negation within SW OWL datasets is that a KB built out of fragments of other KBs, like the Toronto KB mentioned in introduction, is very likely to be consistent, even if it contains intuitively absurd sets of statements, and even if some of the source KBs are inconsistent.

1.2 Specification

The automated weakening of an input KB in order to get rid of undesired consequences has already been studied in the fields of KB debugging and belief revision/contraction (with an emphasis on inconsistent input KBs for the latter). This section introduces a list of formal or informal requirements adopted for the work described in this thesis, which motivate the choices presented in the following chapters, ruling out alternative proposals made in the literature.

1.2.1 Intended meaning

The first (informal) principle followed in this work aims at avoiding the coexistence of two conflicting meanings of a same predicate or constant in the (deductive closure of) the output KB. Let K be the input KB, and ϕ an axiom of K, such that ϕ is still a consequence of the output KB K'. If a is an element of the signature of ϕ (a constant or a predicate), then ideally, there should be no consequence ψ of K' such that the meaning of a in ψ is incompatible with its meaning in ϕ. For instance, consider the 4 following axioms:

Ex 1.2.1 $K = \{$

(1) $\texttt{director}(\textit{Museum of the Rockies}, \textit{Smithsonian Institution})$

(2) $\texttt{director}(\textit{Hannah and her sisters}, \textit{Woody Allen})$

(3) $\exists \texttt{director}.\top \sqsubseteq \texttt{Movie}$

$\forall x, y : \texttt{director}(x, y) \rightarrow \texttt{Movie}(x)$

(4) $\top \sqsubseteq \forall \texttt{director}.\texttt{Person}$

$\forall x, y : \texttt{director}(x, y) \rightarrow \texttt{Person}(y) \quad \}$

And let ϕ be axiom 1. If ϕ is still implied by the output KB K', then no con-

sequence ψ of K' should be such that the meaning of either `director`, *Museum of the Rockies* or *Smithsonian Institution* in ψ is incompatible with its meaning in ϕ. In particular, this rules out some proposals like the one made by [QLB06] for instance (introduced in Chapter 3, Section 3.6.1), and inspired by belief set revision/contraction [AGM85], which, when applied to FOL (or some DLs), may amount to hard-coding exceptions. These views are motivated by minimal information loss expressed as a semantic distance between sets of models (in the model-theoretic sense). Let us assume that one wants to rule out the possibilities that the *Museum of the Rockies* is a `Movie` and that the *Smithsonian institution* is a `Person`, or in other words, `Movie`(*Museum of the Rockies*) and `Person`(*Smithsonian Institution*) have been identified as two undesired consequences of K. Then according to the approach of [QLB06],[14] any KB equivalent to[15] the following would be an admissible solution for example 1.2.1:

$K' = \{$

(1) `director`(*Museum of the Rockies*, *Smithsonian Institution*)

(2) `director`(*Hannah and her sisters*, *Woody Allen*)

(3') $\exists$`director`.$\top \sqsubseteq$ `Movie` $\sqcup \{$*Museum of the Rockies*$\}$

$\forall x, y :$ `director`$(x, y) \rightarrow$ `Movie`$(x) \vee x =$ *Museum of the Rockies*

(4') $\top \sqsubseteq \forall$`director`.`Person` $\sqcup \{$*Smithsonian Institution*$\}$

$\forall x, y :$ `director`$(x, y) \rightarrow$ `Person`$(y) \vee y =$ *Smithsonian Institution* $\}$

Axioms 3 and 4 have been weakened in order to accommodate for axioms 1, 2, and the fact that `Movie`(*Museum of the Rockies*) and `Person`(*Smithsonian Institution*)

[14] Specifically, the approach of [QLB06] applies to the case of revision, i.e. assuming here that K is revised by $\{\neg$`Person`(*Smithsonian Institution*), $\neg$`Movie`(*Museum of the Rockies*)$\}$ understood conjunctively. See Chapter 3 Section 3.6 for an introduction to revision.

[15] The syntactic formulation of K' is not relevant here, as explained in section 1.2.3.

should not hold. The predicate `director` must now have a "movie or the Museum of the Rockies" as its first argument, and a "person or the Smithsonian Institution" as its second argument. K' is strictly weaker than K. In particular, `Movie`(*Museum of the Rockies*) and `Person`(*Smithsonian Institution*) are not consequences of K', as intended. But it still holds that *Woody Allen* directed *Hannah and her sisters* just like the *Smithsonian Institution* directed the *Museum of the Rockies*, such that if ψ is 2 (an axiom is also a consequence), then the meaning of `director` in ψ does not correspond to its meaning in ϕ, which contradicts the above principle.

1.2.2 Consistency and preserved knowledge

As will be seen in Chapter 3, KB debugging and belief set revision/contraction have largely focused on inconsistent input KBs, or on KBs for which a set of undesired consequences is already identified. But as explained in Section 1.1.2, absurd but nonetheless consistent OWL KBs are a common phenomenon. Therefore it is assumed that the input KB may or may not be logically consistent.[16] And that some undesired consequences may have been identified beforehand or not. In other words, the detection of nonsense may be an integral part of the process, and not only the repair.

The output KB on the other hand should be consistent, and if there were undesired consequences, they should not be entailed by it.

Additionally, it may be desirable to preserve some subset Θ of the input KB K during the debugging process, if Θ is considered more reliable than the rest of K. This possibility should be granted, provided Θ is consistent.

[16] or logically coherent

1.2.3 Syntax dependent debugging

Debugging strategies based on belief set revision/contraction are also characterized by their independence from syntax. The input KB K and the output of the process are primarily viewed as (deductively closed) theories, regardless of their syntactic formulations. For instance, let $\phi_1 = A \sqsubseteq B$ (in FOL, $\forall x : A(x) \rightarrow B(x)$), let $\phi_2 = A \sqsubseteq C$, and $\phi_3 = A \sqsubseteq B \sqcap C$ (in FOL, $\forall x : A(x) \rightarrow (B(x) \wedge C(x))$. And let $K_1 = \{\phi_1, \phi_2\}$, and $K_2 = \{\phi_3\}$. According to this view, although they differ syntactically, K_1 and K_2 would be considered as the same input KB.

Another approach to KB debugging, more widespread in the DL community [SC03, Sch05, BP10], requires that the output KB K' be a syntactic subset of the input KB. In this case, K' cannot be identical for K_1 and K_2, unless $K' = \emptyset$. This view may seem counterintuitive, but is often relevant in order to ensure traceability, i.e. to keep track of the original formulation (together with the source) of the axioms of the input KB. It is generally the case for SW datasets, which tend to rely on the integration of axioms imported from other KBs. Therefore this is also the approach followed in this thesis.

Another motivation for this choice, illustrated by example 1.1.1, is the fact that axioms tend to make sense when considered individually, even if the KB as a whole is intuitively absurd. Therefore it may be useful to keep track of the initial syntactic formulation of the discarded knowledge as well, in order to rephrase these axioms accordingly (as mentioned in introduction though, this posterior reformulation falls out of the scope of this work).

An additional and more technical argument for syntax dependent debugging is discussed in Chapter 3. To our knowledge, aside from model-based semantic distances like the one just mentioned in Section 1.2.1, belief set revision (but not contraction)

for certain DLs does not currently provide an effective principle which guarantees minimal information loss. On the other hand, syntax dependent debugging offers a relatively intuitive solution, which is maximality wrt set inclusion among semantically admissible candidate subbases of K.

A final advantage of syntax dependent debugging is that some preference relation over the axioms of K may be used to guide the debugging process. This preference relation may for instance come from different confidence levels attributed to the respective sources of the axioms of K. Or it may come from an external source of knowledge, in particular from linguistic evidence, as illustrated in Chapter 8.

1.2.4 ABox/TBox

Let us focus again on the 5 first axioms of example 1.1.1:

Ex 1.2.2 $K = \{$

(1) $\texttt{keyPerson}(\textit{Caixa Bank}, \textit{CEO})$

(2) $\texttt{keyPerson}(\textit{Sina Bank}, \textit{CEO})$

(3) $\texttt{occupation}(\textit{Peter Munk}, \textit{CEO})$

(4) $\texttt{keyPerson}(\textit{BrookField Office Properties}, \textit{Peter Munk})$

(5) $\top \sqsubseteq \forall \texttt{keyPerson}.\texttt{Person}$

$\forall x, y : \texttt{keyPerson}(x, y) \rightarrow \texttt{Person}(y) \quad \}$

One may choose to consider that facts or so-called *ABox axioms* (see Chapter 2, Section 2.3.7), like 1, 2, 3 and 4, as more reliable than an abstract axiom like 5, called a *TBox axiom*, because assessing a raw fact is arguably less error-prone that coining an abstract property. According to this view, the meaning of `keyPerson` which should prevail is "has as a key person someone whose occupation is", because it is dominant among ABox axioms. Conversely, one may consider that TBox axioms

as more fundamental due to their structuring role for the KB, by analogy to a RDB schema. According to this latter view, the meaning of `keyPerson` which should prevail in example 1.2.2 is "has as a key person".

By default, the choice is made in this thesis to remain agnostic, i.e. to consider that neither the ABox or the TBox should prevail *a priori*. It turns out for instance that both the DBpedia ABox and the DBpedia TBox can be edited (directly or indirectly) by anonymous contributors. But if this is justified by the application, or by the sources of the data, the debugging solutions presented in chapters 6 and 8 are flexible enough to either enforce the preservation of a consistent ABox or TBox, or prioritize the removal of axioms from one of the two.

There is also a more technical argument supporting this view, explained in Chapter 2, Section 2.3.5, which is that the ABox/TBox distinction tends to blur for expressive Descriptions Logics, as soon as so-called *nominals* are allowed.

As a consequence, without further information, either the first or the second meaning of `keyPerson` may prevail in example 1.2.2, and according to the principle introduced in Section 1.2.1, at most one of them can be preserved in the output KB.

1.2.5 Ontological debugging

The term "ontology" has seen an important gain in popularity in knowledge representation in recent years. The most cited definition is probably the one given in [Gru95], where an ontology as a computational artifact is presented as an "explicit specification of a conceptualization". This is unfortunately not formal enough for the work presented here, and there seems to be no consensus among computer scientists about what an ontology concretely is.

For instance, in the DL community, "ontology" is generally a synonym for KB.

Similarly, an ontology is essentially characterized as a (function-free) first-order theory in [GOS09], with a standard model-theoretic semantic for FOL (introduced in Chapter 2 Section 2.3.4), with the additional (informal) requirement that it should be as faithful as possible, i.e. that the actual models of the theory should ideally correspond to the models intended by the KB engineer. But another widespread usage consists in using the term "ontology" to refer to a TBox exclusively,[17] or sometimes only part of it.

An illustration of this terminological ambiguity can for instance be found on the web page describing the DBpedia ontology.[18] On the one hand, the ABox component is presented as part of the ontology ("The DBpedia Ontology currently contains about 4,233,000 instances"), including binary predicates instances (labeled as "DBpedia Ontology other A-Box properties"). On the other hand, in "the ontology is a directed-acyclic graph, not a tree. Classes may have multiple superclasses. . . ", the term ontology apparently refers to part of the TBox, namely its taxonomy. The term "taxonomy" itself is ambiguous. It may designate the set of all statements of a KB of the form $A \sqsubseteq B$ ($\forall x : A(x) \rightarrow B(x)$ in FOL), where A and B are atomic DL concepts, like `Person` or `Organization`. But it may or may not include so-called *disjointness axioms*, i.e. axioms of the form $A \sqsubseteq \neg B$ ($\forall x : A(x) \rightarrow \neg B(x)$ in FOL). Furthermore, it may or may not include formulas of these two types which are not explicitly stated, but are part of the deductive closure of the KB.

This is the reason why the terms "KB" and "TBox", arguably less ambiguous, are preferred in this thesis. But the term "ontological debugging" is nonetheless used, and with a very specific meaning, which needs to be specified. "Ontological" here refers to a more philosophical understanding of the term, used for instance in

[17] or the union of the TBox and the so-called *RBox* of a DL KB, see Chapter 2 Section 2.3.7
[18] `http://wiki.dbpedia.org/services-resources/ontology`

[GG95]. It indicates that the input data are not just factually erroneous, but absurd, and that this is the type of errors targeted by the debugging strategies proposed in the following chapters. In other words, considered as a whole, an input set of statements like example 1.1.1 violates intuitions about reality which may be viewed as necessary, as opposed to contingent ones. For instance, it violates "nothing can be both a person and the occupation of a person", which would probably be viewed by many as a necessary truth, whereas "Peter Munk is a CEO" would rather be viewed as a contingent one. Another illustration is "Woody Allen is a movie director and has an entrance fee", which may be viewed as a necessarily false, as opposed to "Woody Allen directed *Citizen Kane*", which may be viewed as factually false only. This distinction is prototypically the one modeled by epistemic or alethic modal logics, which DLs under standard first-order semantic cannot express. The distinction is also possibly subjective, but a relative consensus is nonetheless often evidenced by natural languages, which is of particular interest for this work. Syntactically correct but semantically ill-formed sentences are a well-attested linguistic phenomenon, notoriously illustrated by Noam Chomsky's "Colorless green ideas sleep furiously". Similarly, many English speakers would consider "during Woody Allen" or "closing Woody Allen" as improbable phrases. On the other hand, picking up a random sentence containing "Orson Welles" from the web and replacing it by "Woody Allen" will probably yield a semantically well-formed sentence, even if factually incorrect.

1.3 Proposals

1.3.1 Distributional evidence

Section 1.2 listed some of the requirements to be met (strictly for some of them, ideally for others) by the debugging strategies proposed in this thesis. But almost no indication has been given yet regarding the effective selection of adequate subset(s) of the input KB.

One of the main contributions of this work is the use of linguistic evidence automatically gathered from web pages for that purpose.[19] More exactly, linguistic evidence allows for the identification of marginal meanings of some constants and predicates within the input KB. The approach is based on relatively simple techniques issued from named entity classification/ontology population, and makes use of distributional semantics (for an introduction to distributional semantics, see [TPo10, Cla13], as well as Chapter 4 section 4.3). The underlying assumption is that individuals which instantiate the same concepts in a consistent/coherent KB also tend to share the same linguistic contexts, and conversely. It can be viewed as a generalization of a well-known linguistic phenomenon named *selectional preferences* [Res97], i.e. the fact that some words (verbs, adjectives, prepositions, ...) tend to select select nouns or noun phrases of a certain semantic or ontological type as their syntactic arguments. For instance, "X was born in" tends to select a human being, whereas "X was launched" does not. The assumption adopted here can be viewed as a generalization of this phenomenon, in that it is not restricted to a small list of abstract semantic types (like `Person`, `Place`, `Event`, ...), but applies to arbitrary

[19] This is not the only investigated possibility. In particular, Chapter 7 focuses on an alternative (or possibly complementary) approach, which is the detection of nonsense on a logical basis, by manually introducing explicit ontological distinctions.

unary predicates of the signature of the input KB. As such, it was successfully used for ontology population by [TM08] or [GG08]. But the proposals made in Chapter 4 Section 4.2 actually go further in that direction, investigating the applicability of this hypothesis to so-called *complex DL concepts* (defined in Chapter 2 Section 2.3.2), more specifically to the complement of an atomic concept (i.e. the complement of a unary predicate in FOL), as well as concepts such as "things which have an author" or "thing that own something".

If Q is a candidate subbase of the input KB K (possibly K itself), and if ψ is a consequence of Q of the form $C(e)$, with C a (possibly complex) DL concept and e a constant, then it should be possible to evaluate whether ψ is likely to hold if the rest of Q does, according to linguistic evidence. As an illustration, consider the 3 following axioms from example 1.2.1:

Ex 1.3.1 $\Delta = \{$

(1) $\texttt{director}(\textit{Museum of the Rockies}, \textit{Smithsonian Institution})$

(2) $\texttt{director}(\textit{Hannah and her sisters}, \textit{Woody Allen})$

(3) $\top \sqsubseteq \forall \texttt{director}.\texttt{Person}$

$\forall x, y : \texttt{director}(x, y) \rightarrow \texttt{Person}(y)$ $\}$

And let us assume that these 3 axioms are part of a larger consistent[20] KB K. Then the following formulas are all consequences of K:

(c_1) $\texttt{Person}(\textit{Smithsonian Institution})$

(c_2) $\texttt{Person}(\textit{Woody Allen})$

(c_3) $\exists \texttt{director}.\top(\textit{Museum of the Rockies})$

$\exists x : \texttt{director}(\textit{Museum of the Rockies}, x)$

(c_4) $\exists \texttt{director}.\top(\textit{Hannah and her sisters})$

[20] and coherent in the DL sense

$$\exists x : \texttt{director}(\textit{Hannah and her sisters}, x)$$

(c_5) $\quad \exists \texttt{director}^{-}.\top(\textit{Smithsonian Institution})$

$$\exists x : \texttt{director}(x, \textit{Smithsonian Institution})$$

(c_6) $\quad \exists \texttt{director}^{-}.\top(\textit{Woody Allen})$

$$\exists x : \texttt{director}(x, \textit{Woody Allen})$$

Now let us assume that there are other instances of `Person` according to K, and that these individuals are human beings, for instance:

$$K \vdash \texttt{Person}(\textit{Margaret Atwood})$$
$$K \vdash \texttt{Person}(\textit{Elizabeth Iorns})$$
$$K \vdash \texttt{Person}(\textit{Peter Munk}) \ldots$$

Then one may expect the linguistic behavior of the linguistic term "Woody Allen" to be relatively similar to the behaviors of terms denoting other instances of `Person` according to K (precisely, more similar than it is to the behavior of a random individual of K). Conversely, one may expect the similarity between "the Smithsonian Institution" and terms denoting other instances of `Person` according to K to be relatively low. For instance, sequences of words like "Woody Allen was born" or "Woody Allen grew up" are more probable than "the Smithsonian institution was born" or "the Smithsonian institution grew up". Therefore, and *ceteris paribus*, `Person`(*Woody Allen*) would be identified as likely to hold if the rest of K holds, and `Person`(*Smithsonian Institution*) as unlikely.

Similarly, let us assume that the second argument of the relation `director` according to K is generally a movie director, for instance:

$$K \vdash \exists \texttt{director}^{-}.\top(\textit{Orson Welles})$$

$K \vdash \exists \texttt{director}^{-}.\top(\textit{Terry Gilliam}) \ldots$

Then again, one may expect the linguistic behavior of "Woody Allen" ("Woody Allen's work", "Woody Allen was born", ...) to be relatively similar to the behavior of terms denoting other instances of $\exists \texttt{director}^{-}.\top$, and that it is not the case for "the Smithsonian Institution".

Finally, let us assume that the first argument of the relation `director` according to K is generally a movie, for instance:

$K \vdash \exists \texttt{director}.\top(\textit{Citizen Kane})$
$K \vdash \exists \texttt{director}.\top(\textit{The meaning of life}) \ldots$

Then again, one may expect the linguistic behavior of "Hannah and her sisters" to be relatively similar to the behavior of terms denoting other instance of $\exists \texttt{director}.\top$ ("Hannah and her sisters was released on", "played in Hannah and her sisters", ...), and that it is not the case for "the Museum of the rockies".

In order to keep the example simple, let us make the following additional assumptions:

- within K, only the 3 axioms of Δ are candidate for removal
- $K \setminus \Delta \nvdash \texttt{Person}(\textit{Smithsonian Institution})$
  $K \setminus \Delta \nvdash \texttt{Person}(\textit{Woody Allen})$
  $K \setminus \Delta \nvdash \texttt{director}.\top(\textit{Museum of the Rockies})$
  etc.

Then according to the requirements listed in Section 1.2, there is only one way Δ can be weakened in order to get rid of the 3 unlikely consequences above (c_1, c_3 and c_5), and retain the 3 likely ones (c_2, c_4 and c_6), namely discarding axiom 1, which is also the intuitive solution.

Several conditions must be met for this approach to be applied. First, it focuses on the contexts of individual constants (an in-depth discussion of this choice is provided in Chapter 4 Section 4.2.1), therefore the input KB's signature should contain individual constants, and they should have linguistic labels (like "Woody Allen" is a linguistic label for the constant *Woody Allen*). A second condition for this approach to be applied is that it should be possible to retrieve with a search engine a sufficient number of web pages where these linguistic labels appear.

1.3.2 Debugging strategies

This form of linguistic evidence in incorporated to several of the debugging strategies proposed (and for some of them evaluated) in Chapter 6 and Chapter 8.

For each candidate subbase Q and each consequence ψ of Q of the form $C(e)$ decribed above, a score $\text{sc}_Q(\psi)$ called a *plausibility score* and based on distributional semantics is computed, which evaluates to what extent ψ is likely to hold if the rest of Q does. These scores are then used to select optimal subbases of K, returned as output of the process.

If the input KB K is consistent, two alternative algorithms are proposed in Chapter 6 Section 6.3.2, which select subbases of K as outputs, based on plausibility scores.

If the input KB K is inconsistent,[21] an original form of base debugging is proposed in Chapter 8, which relies on the previous computation of all minimal conflicts or *justifications* for the inconsistency. For each potentially faulty axiom ϕ of K (i.e. involved in the inconsistency of K), two bases S_ϕ and $S_{\backslash\phi}$ are computed, which represent the part of K which would necessarily be preserved if ϕ was respectively

[21] or incoherent

retained or discarded, according to the minimal information loss principle mentioned in Section 1.2.3, (i.e. that an output subbase must be maximal wrt set inclusion among consistent subbases of K). S_ϕ and $S_{\backslash\phi}$ may in particular be evaluated on a linguistic basis,[22] which yields a preference relation over all candidate axioms for removal. Then a form a prioritized base debugging is performed, which favors the removal of least preferred axioms while respecting this minimal information loss principle.

A third option is also investigated in Chapter 7, which consists in manually extending a consistent input KB with a foundational ontology, introducing fundamental distinction between abstract concepts, in order to yield an inconsistency. The resulting inconsistent KB may then serve as input to the second debugging strategy just introduced, with the additional requirement that the foundational ontology should be preserved during the debugging process.

1.3.3 Evaluation

Two very different categories of input are used for the different evaluations described in the following chapters. The first category consists of sets of logical statements (up to $\approx$ 8000 axioms) automatically extracted from DBpedia.

Such data are likely to contain intuitively nonsensical sets of statements, as illustrated by example 1.1.1, and therefore are relevant application scenarios for these debugging strategies.

But it turns out that manually evaluating the performance of a debugging algorithm on a real dataset is non trivial, as illustrated by example 1.1.1. As noted in Section 1.1.1, three conflicting meanings of `occupation` seem to coexist, and accord-

[22] this is only one possibility

ing to the commitment made in Section 1.2.1, at most one of them should be retained. But it is not obvious for a human evaluator to decide which of these 3 meanings is the correct one, and therefore which axioms should be preferably discarded.

This is why a second and fully automated evaluation protocol was also designed. It takes as input sets of statements of an arguable higher quality (described in Chapter 5, Section 5.2), but automatically degraded, by generating random additional axioms out of their signatures, the underlying assumption being that these random axioms are very likely to be absurd. A debugging strategy is then evaluated based on its ability to automatically spot the randomly generated axioms.

1.4 Content

This section gives an overview of the organization of the thesis. Due its interdisciplinary character, technical reviews of existing works are not all grouped into a single chapter, although most of them are at least introduced in Chapter 3.

In addition, in order to avoid possible misunderstandings, it should be noted that the succession of the different chapters is essentially dictated by readability, and is not meant to reflect a unique global debugging process, neither the chronological order in which the respective contributions were made. More generally, this thesis should be viewed a series of relatively independent and more or less practical/theoretical contributions, all motivated by the same objective, namely the detection and/or repair of nonsense in an OWL KB.

1.4.1 Chapter 2: notation and preliminary notions

This chapter introduces Description Logics, as well as the main notational conventions adopted throughout the thesis.

1.4.2 Chapter 3: KB debugging, state of the art

This chapter is a general overview of KB debugging techniques, with a strong emphasis on algorithms involving some form of reasoning, as well as an introduction to belief change.

1.4.3 Chapter 4: linguistic evidence

This chapter focuses on linguistic aspects. Section 4.1 gives an overview of some of the main paradigms used in information extraction/ontology learning, and discusses their applicability to the problem at hand. The main proposals of this chapter are made in Section 4.2. A so-called *plausibility score* is first defined which, for a given consequence ψ of a candidate output base Q of a debugging process, evaluates to what extent ψ is likely to hold if the rest of Q does, assuming some distributional representations of the linguistic labels of the individuals of the input KB K. Then four alternative preference relations over the family of candidate output bases are defined, based on the plausibility scores of their respective consequences. These preference relations will be used to guide some of the debugging processes introduced in Chapters 6, and to a lesser extent in 8. Finally, Section 4.3 discusses the computation of distributional representations of individual labels, and introduces the specific distributional settings used for the experiments described in the following chapters. Most proposals made in Chapter 4 were published in [CAGV15a].

1.4.4 Chapter 5: datasets

Chapter 5 describes the datasets (and linguistic corpora) used for the experiments of chapters 6, 7 and 8. The choice was made to group the presentation of all datasets in a single chapter, because some of them are used multiple times in the following ones. Two very different types of datasets are presented. Datasets of the first type are automatically extracted subsets of DBpedia (up to several thousands of axioms), which are likely to contain violations of common sense of the kind presented in example 1.1.1. As an alternative, datasets of the second type are arguably more reliable KBs, issued from academic research projects, but automatically degraded for the sake of the different evaluations, i.e. extended with random axioms generated out of their signature, the assumption being that such axioms are very likely to be absurd wrt to the rest of the KB.

1.4.5 Chapter 6: trimming a consistent KB

Chapter 6 investigates the usage of linguistic evidence to debug a consistent input KB K. Section 6.1 evaluates the plausibility score defined in Chapter 4, and section 6.2 evaluates the four preference relations over candidates subbases (defined in Chapter 4 too), but for the case where a single faulty axiom needs to be identified. Both evaluations rely on the automatically degraded datasets presented in Chapter 5.

Section 6.3 addresses the identification of sets of faulty axioms within K. It first discusses the difficulty of computing exactly the optimal subbases according to at least two of the four preference relations defined in Chapter 4. Two algorithms are then proposed, and the first one evaluated with both real and automatically degraded datasets. The third algorithm is a more complex procedure, which yields all optimal subbases of K wrt one of the aforementioned four preference relations. Correctness

is proven, together with the fact that given a finite set Ψ of consequences of K with plausibility scores, worst-case complexity for this problem is identical to that of computing for each $\psi \in \Psi$ the family of maximal subsets of K which do not entail ψ. A small part of the work described in Chapter 6 was published in [CAGV15d] and [CAGV15a].

1.4.6 Chapter 7: introducing explicit ontological distinctions

As an alternative (or complement) to linguistic evidence, Chapter 7 investigates the identification of common sense violations within K on a logical basis, and in particular the possibility of extending K with domain-independent explicit distinctions issued from the field of formal ontology. A shallow manual extension strategy is proposed and evaluated, which relies on an external foundational ontology. The main motivation behind this proposal is to limit the manual cost of the operation, as opposed to deeper but time-consuming methodologies like Ontoclean [GW00]. If K is the resulting inconsistent KB, then among other possibilities, a syntax-based revision process like the ones presented in Chapter 8 may be applied to K as a follow-up to this manual extension.

This work was published in [CAGV15b].

1.4.7 Chapter 8: syntax-based contraction/revision

Chapter 8 focuses on a specific form of KB debugging, which aims at identifying sets of axioms to be preferably discarded from an inconsistent input KB. The problem has been designated in the literature as syntax-based contraction/revision, base contraction/revision, or diagnosis.

Section 8.1 characterizes the problem, and identifies two issues which are inherent to the task, namely the number of candidate output bases (or equivalently the number of diagnoses), and computational cost.

Section 8.2 reviews solutions proposed in the literature in order to address these two issues, showing their respective limitations. The approach adopted to deal with the first issue is known as prioritized base contraction/revision. Intuitively, provided a preference relation $\preceq_a$ over the axioms of the input KB K, it consists in prioritizing the removal of least preferred axioms wrt $\preceq_a$ until consistency is reached, together with discarding minimal sets of axioms wrt to set inclusion.

Section 8.4 proposes an algorithm which performs prioritized base contraction/ revision, provided the family of justifications (defined in Chapter 3 Section 3.5) for the inconsistency is known, but without the need to compute all diagnoses (defined in Chapter 3 Section 3.6.2.2).

Section 8.5 proposes a general framework in order to obtain the preference relation $\preceq_a$, in the form of two bases S_ϕ and $S_{\setminus\phi}$ which, for each candidate axiom ϕ for removal, represent what part of the input KB would necessarily be retained if ϕ was respectively retained or discarded. In particular, it is shown that these two bases can be computed in polynomial time if the justifications for the inconsistency of K are known.

Section 8.6 finally evaluates this strategy, for the specific case where the compliance (as defined in Chapter 4) of S_ϕ and $S_{\setminus\phi}$ to some automatically gathered linguistic input is used to rank candidate axioms for removal.

Part of the proposals made in Chapter 8 were published in [CAGV15c].

Chapter 2

Notation and preliminary notions

2.1 Sets

If X is a set, 2^X will designate the power set of X, and 2^{2^X} will stand for $2^{(2^X)}$, i.e. the power set of 2^X.

2.2 Ordering

If $\preceq$ is a total or partial preorder (transitive, reflexive, but not necessarily antisymmetric) over some finite set Δ, then:

- $\min_{\preceq} \Delta$ (resp. $\max_{\preceq} \Delta$) is the set of minimal (resp. maximal) elements of Δ wrt $\preceq$.
- $\delta_1 \prec \delta_2$ stands for $\delta_1 \preceq \delta_2$ and $\delta_2 \not\preceq \delta_1$.

"min" (resp. "max") may also be used without subscript, and in this case, it should be understood as $\min_{\leq}$ (resp. $\max_{\leq}$).

If $\preceq_a$ is a total preorder over some finite set Δ, then:

- $\sim_a$ is the equivalence relation over Δ induced by $\preceq_a$.
- $\Delta/\sim_a$ denotes the quotient set of Δ by $\sim_a$.
- $\Delta^{\preceq_a} = (\Delta_1^{\preceq_a}, .., \Delta_n^{\preceq_a})$ is the list of ordered elements of $\Delta/\sim_a$, such that $1 \leq i < j \leq n$ iff for all $\delta \in \Delta_i^{\preceq}$, and for all $\delta' \in \Delta_j^{\preceq}$, $\delta \prec \delta'$ holds.

2.3 Description Logics

The prototypical target input KBs for the debugging strategies presented in this work are expressed in OWL, although many proposals apply to other knowledge representation languages as well.

The expression "expressed in OWL" is actually slightly misleading, because OWL is a family of languages, not a single language. OWL is a W3C recommendation since 2004, the latest version (2012) being available at this address: `http://www.w3.org/TR/2012/REC-owl2-overview-20121211/`. Most languages of the OWL family correspond to a Description Logic (DL), and all DL based languages of the OWL family to date are at most as expressive as OWL 2, which corresponds to the DL $\mathcal{SROIQ}^{(\mathcal{D})}$. Therefore this is the DL introduced in this section (provided a slight simplification pertaining to datatypeProperties, explained in Section 2.3.1, such that the presented DL is actually $\mathcal{SROIQ}$, without the $^{(\mathcal{D})}$ superscript). $\mathcal{SROIQ}^{(\mathcal{D})}$ and less expressive DLs are decidable fragments of First Order Logic (FOL).

This upper bound on expressiveness remains a theoretical limitation though. For instance, most current off-the-self libraries to reason with OWL datasets do not support all OWL 2 features. The full expressiveness of OWL 2 / $\mathcal{SROIQ}^{(\mathcal{D})}$ is also rarely used in practice within a single dataset. In particular, none of the datasets

introduced in Chapter 5 makes use of all $\mathcal{SROIQ}^{(\mathcal{D})}$ logical operators (the corresponding least expressive DL is indicated for each of them).

Several syntaxes for languages of the OWL family have been defined, but the choice is made in this thesis to use (whenever possible) the DL notation instead, due to its conciseness.

2.3.1 Datatypes and datatypeProperties

OWL languages support the usage of so-called *datatypes* and *datatypeProperties*. Datatype properties can be viewed as binary predicates whose second argument, or *value*, is an element of a predefined and normalized concrete domain, like the domain of integers, the domain of real numbers, the domain of dates, the domain of strings, etc.

For instance, let us consider the four following axioms:

Ex 2.3.1 $K = \{$

(1) $\texttt{population}(\textit{Virginia}, \text{“96205”})$

(2) $\texttt{releaseDate}(\textit{Hannah and her sisters}, \text{“1986-02-07”})$

(3) $\texttt{director}(\textit{Virginia}, \textit{Woody Allen})$

(4) $\exists \texttt{population}.\top \sqsubseteq \texttt{PpopulatedPlace}$

$\forall x, y : \texttt{population}(x, y) \rightarrow \texttt{PopulatedPlace}(x)$

`director` in axiom 3 is a so-called *objectProperty* in OWL, i.e. it can be understood as a regular FOL binary predicate, whereas `population` and `releaseDate` are datatypeProperties, whose values must be of a certain datatype (integer for `population`, date for `releaseDate`).

Concrete domains (the domain of integers, of dates, ...) are disjoint from the domain of predication in the standard model theoretic semantic for OWL (presented

in Section 2.3.4). Additionally, an element of a concrete domain can only appear as the second argument of a datatypeProperty, and therefore cannot be freely predicated over. For instance, "96205" above can only appear as the value of a datatypeProperty.

Detecting or repairing errors pertaining to datatypeProperty values (for instance the fact that a real number is implausibly high) clearly falls out of the scope of this work, as can be seen from the examples given in Chapter 1 of semantic errors targeted in this thesis. This is why datatypeProperties are treated in what follows as unary predicates (or atomic concepts in the DL terminology), regardless of their value. For instance, in example 2.3.1 axiom 1, the only relevant information for the purpose of this work is the fact that *Virginia* is said to have a population, regardless of its number of inhabitants. Similarly, in axiom 2, the only relevant information is that *Hannah and her sisters* is said to have a release date, regardless of the actual date. So axiom 1 is simply treated as $\texttt{Has}_{\texttt{population}}(\textit{Virginia})$, where $\texttt{Has}_{\texttt{population}}$ is a unary predicate standing for "has a population". Similarly, axiom 2 is treated as $\texttt{Has}_{\texttt{releaseDate}}(\textit{Hannah and her sisters})$. Axiom 4 becomes $\texttt{Has}_{\texttt{population}} \sqsubseteq \texttt{PopulatedPlace}$, or $\forall x : \texttt{Has}_{\texttt{population}}(x) \rightarrow \texttt{PopulatedPlace}(x)$ in FOL. An exception is made for built-in datatypeProperties like `rdfs:comment`, `rdfs:label`, etc. The corresponding axioms are considered as metadata, not as axioms, and are therefore excluded from the logical representation of the KB.

Constraints on admissible values (for instance the fact that it should be an integer superior to 2000) are naturally ignored as well. This is also the case of cardinality constraints on datatypeProperties (like declaring a datatypeProperty to be functional), which become irrelevant, because the identity of two values cannot be verified anymore.

The $^{(\mathcal{D})}$ superscript in $\mathcal{SROIQ}^{(\mathcal{D})}$ stands for the usage of datatypes and datatypeProperties, which is why the DL presented in this section is $\mathcal{SROIQ}$, and not

$\mathcal{SROIQ}^{(\mathcal{D})}$.

2.3.2 Syntax

$\mathcal{SROIQ}$ is a decidable fragment of FOL, and generally considered as a very expressive DL. Non-logical symbols in $\mathcal{SROIQ}$ are of 3 types:

- DL/OWL *individuals* (like *Virginia* or *Woody Allen*), which correspond to FOL individual constants.
- DL *atomic concepts* (like `PopulatedPlace` or `Person`), also called *classes* in OWL, which correspond to FOL unary predicates.
- DL *atomic roles* (like `director`), which correspond to FOL binary predicates. In OWL, these are called *objectProperties*

Most DLs allow for the inductive construction of arbitrarily *complex concepts* out of the atomic ones. For instance, if `LawFirm` and `Accountant` are atomic concepts, and if `worksFor` is an atomic role, then $\texttt{Accountant} \sqcap \exists \texttt{worksFor}.\texttt{LawFirm}$ is a complex concept, which intuitively designates accountants working for a law firm. An atomic concept may also be represented extensively as a set of DL individuals, using so-called *nominals*. For instance, $\{$*Orson Welles*, *Woody Allen*$\}$ is an atomic concept. Extensional representations of concepts can be used in the construction of complex concepts as well, for instance in $\texttt{Movie} \sqcap \exists \texttt{director}.\{$*Orson Welles*, *Woody Allen*$\}$ which would designate movies directed by Orson Welles or Woody Allen.

The construction of complex DL roles is comparatively very limited, restricted to the application of the inverse operator "$.^-$". For instance $\texttt{hasEmployer}^-$ stands for the inverse of `hasEmployer`, i.e. intuitively "is the employer of". Role chains, like $\texttt{hasEmployer} \circ \texttt{hasActivity}$ can also be build, but their only appear in some specific

$$
\begin{array}{lll}
\phi & ::= & \phi_{ABox} \mid \phi_{TBox} \mid \phi_{RBox} \\
\phi_{ABox} & ::= & C(e) \mid R(\mathbf{e},\mathbf{e}) \mid \neg R(\mathbf{e},\mathbf{e}) \mid \mathbf{e} = \mathbf{e} \mid \mathbf{e} \neq \mathbf{e} \\
\phi_{TBox} & ::= & C \sqsubseteq C \\
\phi_{RBox} & ::= & V \sqsubseteq W \mid \mathrm{Dis}(R,R) \mid \mathrm{Asy}(R) \\
C & ::= & \mathbf{A} \mid \top \mid \bot \mid \{N\} \mid \neg C \mid C \sqcup C \mid C \sqcap C \mid \\
 & & \exists R.C \mid \forall R.C \mid \geq_{\mathbf{n}} R.C \mid \leq_{\mathbf{n}} R.C \mid \exists R.\mathrm{self} \\
N & ::= & \mathbf{e} \mid \mathbf{e}, N \\
V & ::= & R \mid R \circ R \\
W & ::= & R \mid U \\
R & ::= & \mathbf{Q} \mid \mathbf{Q}^{-}
\end{array}
$$

Figure 2-1: Syntax of $\mathcal{SROIQ}$

axioms expressing role subsumption, as opposed to atomic roles or their inverse, which may for instance appear in complex concepts. Similarly, role negation ($\neg R$) only appears in some specific formulas (namely in formulas of the form $\neg R(e_1, e_2)$, or (implicitly) in formulas of the form $\mathrm{Dis}(R_1, R_2)$). Therefore, following [HKS06], role chains and negated roles are not considered in what follows as complex DL roles, or in other words, the term *DL role* will refer exclusively to atomic roles and their inverse.

The syntax of $\mathcal{SROIQ}$ is given by figure 2-1 (the semantic of all operators will be given in Section 2.3.4). Terminal non logical symbols are in bold font: **A** for an atomic concept, **Q** for an atomic role, **e** for an individual, and **n** for a positive integers. All other terminal symbols are logical symbols. In particular, $\top$ and $\bot$ are built-in atomic concepts, and U is a built-in atomic DL role, called the *universal role*. "self" is also a logical symbol, with a particular semantic, explained in Section 2.3.4.

This grammar is far from being minimal semantically, in that all symbols are not needed to have the full expressiveness of $\mathcal{SROIQ}$. For instance, $C \equiv D$ can be viewed as syntactic sugar for $C \sqsubseteq D$ and $D \sqsubseteq C$, as well as $\exists R.C$ for $\neg\forall R.\neg C$,

or $\{e_1, \ldots, e_n\}$ for $\{e_1\} \sqcup \ldots \sqcup \{e_n\}$. This non-minimal syntax is preferred here for readability, but also on order to comply to conventions in use in the DL community (in particular, some of these symbols are needed in less expressive DLs).

In order to ensure decidability, some additional syntactic constraints are put on the KB as a whole (and not on each axiom individually). They pertain to role chain characterizations (i.e. axioms of the form $R_1 \circ \ldots \circ R_n \sqsubseteq S$, where $R_1, \ldots, R_n, S$ are DL roles), as well as roles appearing both in role chains and qualified number restrictions (like $\geq_n R.C$). These constraints can be found in [HKS06], but are not detailed here, because they are never discussed in this thesis. It will simply be assumed that an input KB is a valid $\mathcal{SROIQ}$ dataset, i.e. that it verifies these constraints together with the above grammar.

Some other DLs will be mentioned in this thesis, all of which are syntactic subsets of $\mathcal{SROIQ}$. The DL $\mathcal{ALC}$ in particular is one of the most studied in the literature, and is equivalent (under standard first-order model-theoretic semantic, introduced in Section 2.3.4) to the so-called *modal fragment* of FOL. The syntax of $\mathcal{ALC}$ is given by figure 2-2.

The name of a DL generally encodes its expressiveness (although this is not systematic). For instance, $\mathcal{ALC}$ is an extension of the so-called *attributive language* $\mathcal{AL}$ with complex concept negation, represented by the letter $\mathcal{C}$ in $\mathcal{ALC}$. Extending $\mathcal{ALC}$ with atomic role inclusions (i.e. formulas of the form $R \sqsubseteq S$, with R and S atomic DL roles) yields $\mathcal{ALCH}$, where $\mathcal{H}$ stands for atomic role subsumption. Similarly, extending $\mathcal{ALC}$ with nominals (represented by $\{N\}$ in the grammar of figure 2-1) yields $\mathcal{ALCO}$, where $\mathcal{O}$ stands for nominals, and the extension of $\mathcal{ALC}$ with both atomic role inclusions and nominals is $\mathcal{ALCHO}$. The letter $\mathcal{S}$ in $\mathcal{SROIQ}$ is a shortcut for $\mathcal{ALC}$ with transitive roles, such that $\mathcal{SROIQ}$ is actually $\mathcal{ALCROIQ}$ with transitive roles. An exhaustive presentation of the (complex) naming convention

$$\begin{array}{lll} \phi & ::= & C \sqsubseteq C \\ C & ::= & \mathbf{A} \mid \top \mid \bot \mid \neg C \mid C \sqcup C \mid C \sqcap C \mid \exists R.C \mid \forall R.C \end{array}$$

Figure 2-2: Syntax of $\mathcal{ALC}$

scheme of DLs is of little interest for the work presented in this dissertation, and the reader is referred to [BCM+03] for further details.

Another important DL which will be mentioned in this thesis is $\mathcal{SHOIN}$, less expressive than $\mathcal{SROIQ}$, but still considered a very expressive DL. $\mathcal{SHOIN}$ underlies the OWL DL language, which can be viewed as a former version OWL 2, before the OWL 2 specification was adopted. Other important (families of) DLs are $\mathcal{EL}$ and *DL-Lite*, underlying the OWL 2 *profiles* OWL 2 EL and OWL 2 QL respectively. Both are tractable (families of) DLs, the first one mostly used for building terminologies, and the second one tailored for query answering.

2.3.3 Signature

If X is either a DL individual, concept, role, formula, or a set of DL formulas, then $N_{Ind}(X)$, $N_{Con}(X)$ and $N_{Role}(X)$ will designate respectively the sets of DL individuals, atomic concepts, and atomic roles appearing in X. For instance, if $C = \texttt{Accountant} \sqcap \exists \texttt{worksFor}.\texttt{LawFirm}$, then $N_{Ind}(C) = \emptyset$, $N_{Con}(C) = \{\texttt{Accountant}, \texttt{LawFirm}\}$, and $N_{Role}(C) = \{\texttt{worksFor}\}$.

If $N_{Ind}(X)$, $N_{Con}(X)$ and $N_{Role}(X)$ are mutually disjoint, then $\text{sig}(X) = N_{Ind}(X) \cup N_{Con}(X) \cup N_{Role}(X)$ will be called the *signature* of X. By default, i.e. unless explicitly, it is assumed for any concept, role, formula, or a set of DL formulas X considered in what follows that $N_{Ind}(X)$, $N_{Con}(X)$ and $N_{Role}(X)$ are mutually disjoint.

A signature S_1 will be said to be a *subsignature* of a signature S_2 iff the sets

$$
\begin{array}{l}
\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} = \emptyset \\
\neg C^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} = \{x \mid \text{ there is a } y \text{ such that } < x, y > \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \\
(\exists R.\,\text{self})^{\mathcal{I}} = \{x \mid \ < x, x > \in R^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} = \{x \mid \text{ for all } y \text{ such that } < x, y > \in R^{\mathcal{I}},\ y \in C^{\mathcal{I}}\} \\
(\geq_n R.C)^{\mathcal{I}} = \{x \mid |\{y \mid \ < x, y > \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}| \geq n\} \\
(\leq_n R.C)^{\mathcal{I}} = \{x \mid |\{y \mid \ < x, y > \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}| \leq n\}
\end{array}
$$

Figure 2-3: Standard model-theoretic semantic for $\mathcal{SROIQ}$ concepts

of individuals, atomic concepts and atomic roles of S_1 are subsets of the sets of individuals, atomic concepts and atomic roles of S_2 respectively.

2.3.4 semantic

The standard model-theoretic semantic for DLs (also called *descriptive semantic* in [BCM+03]) is the same as for FOL.

Let $\text{sig}(X) = N_{Ind}(X) \cup N_{Con}(X) \cup N_{Role}(X)$ be a DL signature. Then an *interpretation* $\mathcal{I}$ for $\text{sig}(X)$ is a pair $< \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$, where $\Delta^{\mathcal{I}}$ is a nonempty set called the *domain* of $\mathcal{I}$, and $.^{\mathcal{I}} : \text{sig}(X) \mapsto (2^{\Delta})^n$ is an *interpretation function*, with $1 \leq n \leq 2$. For each $e \in N_{Ind}(X)$, $e^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. For each $A \in N_{Con}(X)$, $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$. And for each $R \in N_{Role}(X)$, $R^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^2$.

The unique name assumption is not made by default in OWL, therefore it is not made here either, i.e. if $\{e_1, e_2\} \subseteq N_{Ind}(X)$, $e_1 \neq e_2$, and $\mathcal{I}$ is an interpretation for $\text{sig}(X)$, then it may be the case that $e_1^{\mathcal{I}} = e_2^{\mathcal{I}}$.

The interpretation of complex concepts is then build inductively as in table 2-3, and the interpretation of roles, role negations and role compositions as in table 2-4.

$U^{\mathcal{I}} = (\Delta^{\mathcal{I}})^2$
$(R^-)^{\mathcal{I}} = \{< x, y > \mid < y, x > \in R^{\mathcal{I}}\}$
$(V_1 \circ V_2)^{\mathcal{I}} = V_1^{\mathcal{I}} \circ V_2^{\mathcal{I}}$ (where $\circ$ on the right-hand side stands for the composition of two relations)

Figure 2-4: Standard model-theoretic semantic for $\mathcal{SROIQ}$ roles

Let:
e, e_1, e_2 be DL individuals,
C, C_1, C_2 be DL concepts,
and $R, R_1, \ldots, R_m, R_n$ be DL roles.
Then:
$\mathcal{I} \models C(e)$ iff $e^{\mathcal{I}} \in C^{\mathcal{I}}$
$\mathcal{I} \models R(e_1, e_2)$ iff $< e_1^{\mathcal{I}}, e_2^{\mathcal{I}} > \in R^{\mathcal{I}}$
$\mathcal{I} \models \neg R(e_1, e_2)$ iff $< e_1^{\mathcal{I}}, e_2^{\mathcal{I}} > \notin R^{\mathcal{I}}$
$\mathcal{I} \models C_1 \sqsubseteq C_2$ iff $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
$\mathcal{I} \models R_1 \sqsubseteq R_2$ iff $R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
$\mathcal{I} \models R_1 \circ \ldots \circ R_m \sqsubseteq R_n$ iff $R_1^{\mathcal{I}} \circ \ldots \circ R_m^{\mathcal{I}} \subseteq R_n^{\mathcal{I}}$
$\mathcal{I} \models \mathrm{Dis}(R_1, R_2)$ iff $R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}} = \emptyset$
$\mathcal{I} \models \mathrm{Asy}(R)$ iff for all $< e_1, e_2 > \in R^{\mathcal{I}}$, $e_1^{\mathcal{I}} \neq e_2^{\mathcal{I}}$

Figure 2-5: Standard model-theoretic truth conditions for $\mathcal{SROIQ}$

If ϕ is a $\mathcal{SROIQ}$ formula, and $\mathcal{I}$ an interpretation for a signature S such that $< N_{Ind}(\phi), N_{Con}(\phi), N_{Role}(\phi) > \subseteq N_{Ind}(S) \times N_{Con}(S) \times N_{Role}(S)$, then $\mathcal{I}$ *verifies* ϕ, also written $\mathcal{I} \models \phi$, iff the truth conditions for ϕ described in table 2-5 are satisfied by $\mathcal{I}$. For instance, if K is a KB and ϕ an axiom of K of the form $C(e)$, with C a (possibly complex) DL concept and $e \in N_{Ind}(K)$, then $\mathcal{I}$ *verifies* ϕ iff $e^{\mathcal{I}} \in C^{\mathcal{I}}$.

$\mathcal{I}$ is called a *model* of K iff for all $\phi \in K$, $\mathcal{I} \models \phi$ holds.

The set of all models of a set Γ of formulas will be denoted with mod(Γ)

2.3.5 Consistency and coherence

A KB is said to be *consistent* if it has at least one model.

Another desirable property of a KB, often used in the DL literature, is *coherence*. A DL concept C is said to be *satisfiable* wrt to a KB K iff there is a model $\mathcal{I}$ of K such that $C^{\mathcal{I}} \neq \emptyset$. Then a KB K is *coherent* iff for all $A \in N_{Con}(K)$, A is satisfiable.

In particular, a KB can be consistent and incoherent.[1]

2.3.6 Entailment

If Γ is a set of DL formulas, and ψ a DL formula, then ψ is a consequence of Γ, noted $\Gamma \vdash \psi$ iff all models of Γ are models of $\{\psi\}$. The notation $\psi \in \mathrm{Cn}(\Gamma)$ will also be used for $\Gamma \vdash \psi$. If S is a signature, then $\mathrm{Cn}_S(\Gamma)$ will designate the set of all consequences of Γ whose signature is a subsignature of S.

If Γ is a set of DL formulas, then $\Gamma \vdash \top \sqsubseteq \bot$ will sometimes be abbreviated as $\Gamma \vdash \bot$. This should be without ambiguity, as $\bot$ is not a syntactically valid DL formula. This convention will also be used for other logics, i.e. is Γ is a set of formulas in a given logic $\mathcal{L}$, then $\mathrm{Cn}(\Gamma) = \mathrm{Cn}(\mathcal{L})$ will sometimes be abbreviated as $\Gamma \vdash \bot$.

A *theory* is a set of formulas closed under consequence, i.e. Γ is a theory iff $\Gamma = \mathrm{Cn}(\Gamma)$.

Like FOL or propositional logic, DLs are Tarskian, i.e. given a DL signature, if $\mathcal{L}$ is the set of formulas which can be built over this signature, then for all $\Gamma, \Gamma_1, \Gamma_2 \in 2^{\mathcal{L}}$, the following intuitive properties hold:

(inclusion) $\Gamma \subseteq \mathrm{Cn}(\Gamma)$

[1] It may also be inconsistent but coherent in the limit case where $N_{Con}(K) = \emptyset$

(iteration) $\quad \mathrm{Cn}(\Gamma) = \mathrm{Cn}(\mathrm{Cn}(\Gamma))$

(monotonicity) $\quad$ if $\Gamma_1 \subseteq \Gamma_2$, then $\mathrm{Cn}(\Gamma_1) \subseteq \mathrm{Cn}(\Gamma_2)$

On the other hand, [Rib13] identified some properties which may or may not be verified by a Tarskian and compact logic, and appear to play an important role for the applicability of the classical postulates for belief change, introduced in Chapter 3. Some of these properties, like complementarity, are not verified by most DLs of interest here.

2.3.7 TBox/ABox

A DL KB is traditionally partitioned into 2 subsets (or 3, depending on the authors). The *ABox* of a DL KB is the set of all axioms derivable from the ϕ_{ABox} non terminal symbol in the grammar of table 2-1. Intuitively, the ABox is the assertional part of the KB, expressing facts about individuals. The *TBox* is the terminological part of the KB, which characterizes its predicates. It is often defined as the complement of the *ABox* in the KB, or, for some authors, as the set of all axioms derivable from the ϕ_{TBox} non terminal symbol in the grammar of table 2-1. In this latter view, the remaining axioms (i.e. the axioms derivable from the ϕ_{RBox} symbol) form the so-called *RBox*.

It should be noted though that for DLs with nominals (i.e. allowing extensionally defined atomic concepts, like $\{\mathit{Woody\ Allen}\}$), this distinction between assertional and terminological knowledge becomes inadequate, because an ABox assertion can be expressed with a semantically equivalent TBox axiom. For instance, the formulas $C(e), e_1 \neq e_2$ and $R(e_1, e_2)$, can be respectively translated into $\{e\} \sqsubseteq C, \{e_1\} \sqsubseteq \neg\{e_2\}$ and $\{e_1\} \sqsubseteq \exists R.\{e_2\}$. The converse is not true though, so the ABox may be more accurately defined as the set of axioms of the input KB which can be expressed as

formulas derivable from ϕ_{ABox}. But this definition is still not completely satisfying. For instance, the TBox axiom $\{e_1\} \sqsubseteq \{e_2, e_3\}$ expresses the fact that a least e_2 or e_3 is identical to e_1. This is arguably knowledge about individuals, and there is no equivalent ABox formulation.

Similarly, the distinction between TBox and RBox tends to blur for very expressive DLs. For instance, as noted by [HKS06], the formula $\exists R.\top \sqsubseteq \exists R.\,\mathrm{self}$ expresses a form of reflexivity for the role R, but it is syntactically part of the TBox.

By default, i.e. unless explicitly stated, in the rest of this thesis, the following syntactic convention will be used:

- The ABox of a KB K will designate all axioms of K derivable from the symbol ϕ_{ABox} in the grammar of table 2-1.
- The TBox of a KB K will designate all axioms of K derivable either from the symbol ϕ_{TBox} or from the symbol ϕ_{RBox} in the grammar of table 2-1.

2.3.8 Reasoning

An in-depth description of reasoning tasks and corresponding algorithms for DLs would be off-topic in this thesis. Section 2.3.8.1 simply introduces generic but useful considerations on the subject. Section 2.3.8.2 sketches the execution of a standard tableau algorithm for consistency in DLs, which played a prominent role in the development of expressive DLs. A basic understanding of tableau algorithms will also be required in chapters 3 and 8, when so-called *glass-box* debugging algorithms will be discussed.

2.3.8.1 Tractable and intractable DLs

Standard reasoning tasks in expressive DLs are typically intractable in the worst case. For instance, satisfiability is PSpace-complete in the well studied DL $\mathcal{ALC}$ (which is a notational variant of the modal fragment of FOL, or equivalently of the multi-modal logic K_n). And for $\mathcal{SROIQ}$, it is NExpTime-hard.

In practice though, algorithms developed for these tasks tend to behave relatively well on most inputs. These empirical considerations have led to the development of very expressive DLs in the recent years, guided by decidability more than tractability, resulting in the adoption of $\mathcal{SROIQ}^{(\mathcal{D})}$ as the logic underlying OWL 2 (and previously $\mathcal{SHOIN}^{(D)}$ as underlying the language called OWL DL). Off-the-shelf reasoning libraries can be found for a variety of tasks in different fragments of $\mathcal{SHOIN}^{(D)}$ / OWL 2, which behave relatively well for moderately large datasets.

Concurrently, other works have focused on tractable DLs, most notably the *DL-Lite* [CDGL+05] and $\mathcal{EL}$ [BBL08] families of logics. The former is tailored for query answering, and underlies the OWL 2 QL profile, whereas the latter is mostly used to build terminologies, and underlies the OWL 2 EL profile.

2.3.8.2 Tableau algorithm

The canonical purpose of a tableau algorithm for DLs is to decide satisfiability of a (possibly complex) DL concept, or consistency of a set of axioms. From a more theoretical perspective, termination of a tableau algorithm has also been used to prove decidability of several problems in very expressive DLs. For readability purposes, the choice is made here to focus on the case of consistency. The execution of a tableau algorithm to decide consistency of a set Φ of axioms can intuitively be viewed as an attempt to build a model for Φ. This section provides a relatively high-level and in-

formal description of the execution of such an algorithm. For a more comprehensive introduction, the reader is referred to [BCM+03].

A tableau algorithm to decide the consistency of a set Φ of axioms updates a family $\mathcal{W}$ of sets of DL formulas, such that at each step, Φ is consistent iff at least one $W \in \mathcal{W}$ is consistent. The successive states of the family $\mathcal{W}$ will be designated with $\mathcal{W}_0, .., \mathcal{W}_n$, such that $\mathcal{W}_n$ is the state of $\mathcal{W}$ after termination of the procedure.

A set of so-called *expansion rules* is iteratively applied to the elements of $\mathcal{W}$ throughout the execution. For instance, if there is a $W \in \mathcal{W}_i$ for some $i \in \{0, .., n\}$ such that $A(e) \in W$ and $A \sqsubseteq C \in W$ but $C(e) \notin W$, then W can be expanded, i.e. replaced in $\mathcal{W}$ by $W' = W \cup \{C(e)\}$, or in other words $\mathcal{W}_i$ becomes $\mathcal{W}_{i+1} = (\mathcal{W} \setminus \{W\}) \cup \{W'\}$. Some rules may cause W to be replaced by two new sets of formulas. For instance, if $C_1 \sqcup C_2(e) \in W$ but $C_1(e) \notin W$ and $C_2(e) \notin W$, then W can be replaced in $\mathcal{W}$ by $\{W', W''\}$, with $W' = W \cup \{C_1(e)\}$, and $W'' = W \cup \{C_2(e)\}$. Fresh individuals may also be introduced in the expansion of W, for instance if W contains the formula $\exists R.\top(e)$, but there is no e' such that $R(e, e') \in W$.

A possible set of expansion rules for $\mathcal{ALC}$ will be given[2] in Chapter 8 Section 8.3.2, together with an example of execution.

If $W \in \mathcal{W}_i$ for some $i \in \{0, 1, .., n\}$, let expans$^*(W)$ denote the family of sets of axioms composed of W and all expansions transitively derived from W during the execution, i.e. expans$^*(W)$ is composed of W itself, the expansion(s) of W, the expansion(s) of its expansion(s), etc. A tableau algorithm for DLs generally relies on *blocking* conditions in order to ensure that expans$^*(W)$ is finite. Otherwise, some expansion rules may be applied an infinite number of times, due to cycles in the terminology and existential quantification, for instance in the case where

[2] for a slightly more complex type of tableau algorithm than the one described here though, with a so-called *tracing* mechanism

$W = \{A(e), A \sqsubseteq \exists R.B, B \sqsubseteq \exists S.A\}$.

A *clash* in generally defined as a minimal set of formulas which instantiates an element of a set of pre-identified clash patterns. Prototypical clash patterns are $\{A(e), \neg A(e)\}$, with A an atomic concept and e an individual, or $\{\bot(e)\}$. These are also the two clash patterns considered in the following discussions about tableau algorithms, in particular in Chapter 3 Section 3.5.1 and Chapter 8 Section 8.3.2. But other clash patterns may be used, depending on the implementation and the DL under consideration.

Let Γ be defined by $\Gamma = \{\top(e) \mid e \in N_{Ind}(\Phi)\}$ if $N_{Ind}(\Phi) \neq \emptyset$, and $\Gamma = \{\top(e')\}$ otherwise, with e' a fresh DL individual. After a syntactic normalization of each axiom in Φ, the algorithm may be initiated with $\mathcal{W}_0 = \{\Phi \cup \Gamma\}$. If at any moment during the execution there is a $W \in \mathcal{W}$ which cannot be expanded and contains no clash, then a model $\mathcal{I}$ of Φ can immediately be build out of the atoms of W. The other termination condition is the case where all elements of $\mathcal{W}$ contain a clash. In this case, Φ is inconsistent.

For very expressive DLs (such as $\mathcal{SROIQ}$) with an RBox, a preliminary extension of Φ with some consequences of Φ may be required, like the one described in [HKS06].

2.4 Typographical conventions

2.4.1 Predicate names, individual names and linguistic labels

Predicate and constant names in OWL are normalized strings known as *IRIs*, for "Internationalized Resource Identifiers" (which are themselves extensions of *URIs* or "Uniform Resource Identifiers" to Unicode characters).

For instance, `http://dbpedia.org/ontology/director` is an atomic DL role in the signature of the DBpedia ontology.
Similarly, `http://dbpedia.org/Smithsonian_Institution` is a DL individual in the signature of DBpedia.

If there is no ambiguity, a shorter notation is adopted here, such that the two above IRIs will be written `director` and *Smithsonian Institution* respectively.

Moreover, DL atomic concepts names (like `Person` or `Movie`) and atomic role names (like `director`) are written with a teletype font,[3] the former with a capital first letter and the latter without, whereas individuals names (like *Smithsonian Institution*) are in italics.

Linguistic terms (like "the Smithsonian Institution", "the Smithsonian", ...) are quoted. Note that *Woody Allen* and "Woody Allen" are two different objects, respectively a DL individual name and a linguistic label associated to that individual name ("W. Allen" for instance could be another linguistic label associated with *Woody Allen*).

2.4.2 Variables

Whenever possible, the following typographical conventions for variables are followed :

- A and B for DL atomic concepts
- C and D for possibly complex DL concepts
- R and S for DL atomic roles[4]

[3] This is not the case for variables representing DL concepts and roles, as explained in Section 2.4.2.

[4] R is also used in Chapter 8 for consistent subbases of the input KB.

- e for an individual
- ϕ and ψ for DL formulas. Additionally, given a KB K, ϕ will be preferred for an axiom of K, and ψ for a consequence of K.[5]
- Capital Greek letters ($\Phi, \Psi, \Gamma, \Theta, K, \ldots$) by default for sets of formulas. But also capital Latin letters in some cases, when the set of formulas is an element of a family of set of formulas (see the following point).
- Capital cursive Latin letters (e.g. $\mathcal{D}$) for families of sets of formulas. In this case, the corresponding capital Latin letters (e.g D) are used for the sets which are elements of this family (e.g. $\mathcal{D} = \{D_1, .., D_n\}$).

[5] Axioms of K are consequences of K, but the converse does not hold in general.

Chapter 3

Knowledge base debugging: state of the art

This chapter is an overview of existing Knowledge Base (KB) debugging techniques. It is by no means an exhaustive state of the art, but a selection of approaches which are either related to the proposals made in the following chapters, or relevant to the specific problem addressed in this thesis, namely the coexistence of incompatible meanings of a predicate or constant in an input KB. In particular, an important part of this review is dedicated to debugging techniques which involve some form of reasoning, with an emphasis on Description Logics (DL). It is also a relatively high-level introduction. The choice is made to provide more technical and/or critical reviews of existing approaches in the relevant chapters, for readability.

3.1 Typology

A strict classification of existing KB debugging strategies is difficult to establish. Some useful distinctions are nonetheless listed in this section, but without necessarily referring to existing works, which will instead be reviewed in sections 3.2 to 3.6. The reason is that none of these distinctions provides a clear-cut partition of the field, i.e. for each of them, there is at least one approach in the literature exemplifying both aspects.

Therefore the following state of the art is not organized according to these distinctions. They are instead used locally in order to give a better understanding of some approaches from the literature. These distinctions are also helpful for positioning the proposals made in this thesis.

3.1.1 Strengthening/weakening

A first methodological distinction can be made between approaches which focus on strengthening the input KB, and the ones which focus on weakening it.

At first sight, KB strengthening (i.e. extending a KB with additional knowledge) may seem irrelevant to the problem considered here. Indeed, a large number or approaches which aim at completing a KB do not pertain to debugging. This is the case in particular of techniques inspired by Inductive Logic Programming or Formal Concept Analysis, which focus on the acquisition of TBox axioms. Another example is knowledge extraction/ontology learning from texts (discussed in Chapter 4, but for other reasons).

But a slightly different view on the problem, inspired by database modeling, considers the TBox of a KB (or only part of it) as a set of constraints on the rest

of the KB, by analogy to a data schema in a relational database. For instance, the TBox $\Gamma = \{\top \sqsubseteq \forall \texttt{receivesAward}.\texttt{Award}, \texttt{Award} \sqsubseteq \neg \texttt{Organization}\}$ prevents the usage of the `receivesAward` role with an organization as a second argument.

This is not the only function of a TBox, which may also be used to infer data for instance. The analogy between ABox/TBox on the one hand and data/data schema on the other hand is not technically accurate either. First, negation can also appear in the ABox, even for moderately expressive DLs. And more generally, as explained in Chapter 2 Section 2.3.7, the distinction between ABox and TBox tends to blur for expressive DLs. This analogy is nonetheless a widespread one in the knowledge representation community, and can be enforced to a certain extent by additional syntactical constraints on the KB (for instance giving a name to each extensively defined concept, banning concept negation in the ABox, ...).

So in the case where it is viewed as a data schema, the TBox may be not be constraining enough for applicative requirements, which is the reason why strengthening the TBox may be viewed as a form of debugging. In a similar vein, meta-modeling generally amounts to adding second order constraints on a KB (prototypically on the TBox).

For the work presented in this dissertation, the literature on KB weakening is clearly more relevant than the one on KB strengthening, but as will be discussed in Chapter 7, a preliminary extension phase may be a useful step in the process, in particular for loosely constrained KBs, i.e. from a logical point of view for KBs where negation is sparsely used.

The line between weakening and strengthening is also not always clear for semi-automated approaches to debugging, where the goal is to rewrite some axioms, and therefore both operations are involved.

3.1.2 Automating detection and repair

Most of the following approaches involve a part of manual correction, but the degree to which repair is automated may vary. At one end of the spectrum lies for instance inconsistency detection. It is certainly useful to know that a KB is inconsistent, but this provides almost no information as to how it can be optimally modified in order to restore consistency. At the other end of the spectrum, some typos, redundancies or violations of syntactic conventions can be repaired in a fully automated fashion.

What is meant by repair may also vary. In particular, diagnosis techniques (see Section 3.6.2.2 below) produce as an output (sets of) statements to be preferably discarded to get rid of an error. But no suggestion is made as to how these statements could reformulated, i.e. how the axioms intended by the KB author(s) may be recovered. This is also the case of most of the strategies developed in this thesis, the reformulation of faulty statements being left as a following debugging step.

3.1.3 Syntactic/semantic verifications

The errors to be spotted may be of a semantic or syntactic nature. A syntactic error here is understood as an error whose correction does not affect the semantic of the KB, or in other words yields an equivalent theory. Among those, one may mention redundant assertions (e.g $A \sqsubseteq B$ and $A \sqsubseteq B \sqcap C$), erroneous or missing linguistic label for an individual or atomic concept, or non-compliance with good engineering practices which do not affect the semantic of the KB, such as the ones given in [Rec03].

This should be distinguished from syntactic patterns which have empirically been identified as likely to indicate a semantic error. For instance, in OWL, defining a binary relation R as its own inverse, using the `owl:inverseOf` construct ($R \equiv R^-$

in DL), is considered in [PVSFGP12] as likely to indicate an error. There is indeed a good reason to think that the author's intention was not to express that R is symmetric: a specific OWL construct `owl:symmetric` ($R \sqsubseteq R^-$ in DL) already exists for that purpose. Similarly, the same authors identified that a cycle in the subsumption relation involving more that 2 concepts ($A_1 \sqsubseteq A_2 \sqsubseteq A_3 \sqsubseteq A_1$) was likely to indicate a semantic error. These are considered as cases of semantic debugging, because the purpose of such patterns is the detection of underlying semantic mistakes.

As already mentioned, the proposals made in the following chapters exclusively focus on semantic errors, i.e. errors which have an impact of the possible interpretations of a KB.

3.1.4 Abstract characterization/concrete debugging

As will appear in Section 3.6.1, a large part of the literature on belief change focuses on abstract characterization of operations on KBs viewed as deductively closed theories. In many cases, no practical implementation of these operations is provided (at least outside of propositional logics). On the more concrete side, an important number of works in the DL community provide algorithms and complexity bounds for different debugging tasks, but without empirical evaluation. Finally, some approaches are primarily guided by empirical considerations, not only in terms of computational cost, but also qualitatively.

The different proposals made in this thesis pertain to the two latter categories.

3.2 Reviewing models

A first family of debugging strategies rely on a review of some of the models (in the model-theoretic sense, see Chapter 2 Section 2.3.4) of the input KB. These approaches pertain to TBox strengthening, as explained above in Section 3.1.1. Intuitively, the objective is to help the modeler identify missing constraints in the input KB, which is particularly relevant for datasets where negation is sparsely used.

This may for instance address a relatively widespread misunderstanding of the `rdfs:domain` and `rdfs:range` constructs, noted among others by [RDH+04]. If R is a DL atomic role and C a DL concept, the `rdfs:range` construct can be used to express the DL formula $\phi = \top \sqsubseteq \forall R.C$. Let K be a consistent KB such that $K \nvdash \phi$, $K \nvdash \neg C(e_2)$, but $K \vdash R(e_1, e_2)$ and $K \vdash A(e_2)$, with e_1, e_2 two DL individuals and A an atomic DL concept. Then a common modeling mistake is to consider that $K \cup \{\phi\}$ must be inconsistent, assuming that $K \vdash \neg C(e_2)$ holds by default. This could be the case for instance if $A \sqsubseteq \neg C$ was a statement of K. In other words, in the absence of explicit negation in K, the `rdfs:domain` and `rdfs:range` constructs should not be viewed as the type constraints used in a database schema.[1].

[FR12] developed a tool in order to assist the manual strengthening of a TBox. The idea is to present to the user some ABox statements which are admissible assertions wrt the current state of the TBox of the input KB K. Then if the user identifies among these assertions a formula γ which should be ruled out, the system suggests a stronger version K' of K, such that $K' \cup \{\gamma\}$ is inconsistent. This may be viewed as a form of manual model checking, where γ represents a property verified by some of the models of K (or intuitively some possible worlds according to K),

[1] unless the closed-world assumption is made, which does not correspond to the standard semantic for OWL

and the strengthening phase aims at ruling out these interpretations.

A comparable approach was implemented by [BGBA10], as part of the OntoUML editor, in order to assist the conception of database schemas in UML. The tool is based on the generation of visual representations of minimal or almost minimal models of the current schema, viewed as a logical theory. The user can then rule out the non-admissible interpretations by strengthening the KB.

3.3 Formal ontology

Formal ontology provides axiomatizations of philosophically grounded principles which, among other applications, may be used to identify probable mistakes within a KB. One of the most influential examples of such a debugging strategy is the Ontoclean methodology, introduced in [GW00].

A review of the Ontoclean methodology and its implementations is provided in Chapter 7, followed by a series of experiments which are similar in spirit, but rely on top-level ontology instead, allowing for the introduction of negations in loosely constrained datasets with a very limited amount of manual work.

3.4 Syntactic patterns

Another family of DL KB debugging strategies are characterized by a strong empirical focus, with an emphasis on tractability. They rely on the identification within an input KB of syntactic patterns which are likely to indicate a modeling mistake, without requiring the use of a full-fledged reasoning service. In particular, the input KB may or may not be consistent/coherent.

Some representative works are the OOPS! validation tool [PVSFGP12], and the

"antipatterns" by [RZ13] or [SBG12]. An antipattern is a syntactic pattern likely to indicate an error. Some of them are related to the influential work of [RDH+04] on the identification of mistakes commonly made by occasional users of logic-based knowledge representation languages. The errors which may be spotted are extremely diverse, even for a same tool. As a consequence, a more detailed description of these approaches is difficult without an exhaustive enumeration of the syntactic patterns, which would be irrelevant here. In particular, some of the errors being spotted have nothing to do with the logical formulation of the KB. For instance, [PVSFGP12] allows for spotting irregular naming conventions (capital letters, underscore, ...) in the signature of a KB, as well as missing annotations or linguistic labels (which are considered as meta properties of the input KB, and not part of its logical formulation). Another example is the (second-order) declaration of an atomic concept A (for instance stating that `Person` is an atomic concept) when this concept is not used in any (first-order) logical statement of the KB, identified as a mistake in [RZ13] and [PVSFGP12].

For semantic mistakes now, the patterns may either spot missing information (indicating that the KB is too weak), or point at incorrect sets of statements (indicating that the KB is too strong). The absence of explicit domain and range for a DL role, spotted by [PVSFGP12], is an example of probably missing information. An example of a probably incorrect set of statements, still in [PVSFGP12], consists in declaring two roles as each other's inverse when the declared domain of the former is not identical to the declared range of the latter. It should be noted though that a KB containing this pattern may still be consistent/coherent.

More generally, debugging patterns are based on what may be viewed as a generalized notion of typo. Although violations of such patterns may indeed indicate semantic errors, errors of the type introduced in Chapter 1 (namely the coexistence

of multiple meanings of a predicate or individual within a KB) usually go unnoticed when such patterns are applied, especially for datasets with a relatively low expressiveness. For instance, in example 1.1.1, each formula is syntactically very basic (of the form $A \sqsubseteq B, A(e), R(e_1, e_2), \top \sqsubseteq \forall R.A$, ..). And by using a different signature, it is actually easy to find a replacement for each element of $\text{sig}(K)$ in this example, such that the whole set of statements after replacement is not absurd anymore. As an illustration, let us replace each individual in example 1.1.1 which does not denote a human being by a human being: for instance *Caixa Bank* becomes *Margaret Atwood*, *CEO* becomes *Woody Allen*, etc. Let us also replace each DL role (`keyPerson`, `occupation`, ...) by a relation which may hold between two human beings, like `knows`, `worksWith`, or `hasRelative`, and let us replace the DL atomic concepts `PersonFunction` and `Organization` by `LivingThing` and `Agent` respectively. The resulting set of statements could be factually wrong (maybe Margaret Atwood and Woody Allen do not know each other), but it would not contain violations of common sense. And from a syntactic point of view, it is identical to example 1.1.1. This shows that no possible syntactic pattern is able to spot the nonsense in example 1.1.1.

The patterns used by [SBG12] are slightly different, in that they are not only based on common mistakes or syntactic typos, but also grounded philosophically. These patterns identify violations of (second-order) constraints formalized by the foundational ontology UFO [Gui05]. The input KB before debugging must already comply to the OntoUML specification[Gui05] though, which is a UML profile based on UFO. Therefore these patterns do not apply in general to standard OWL KBs, falling out of the scope of this work.[2].

[2]Providing an example of such a pattern here would require the introduction of the specific philosophical terminology used in OntoUML, which also falls out of the scope of this work.

Finally, the degree to which repair can be automatized varies with the type of errors being spotted. Non-compliance to syntactic conventions can sometimes be repaired automatically. But for patterns indicating a possible semantic mistake (see section3.1.3), the repair may be a more complex and ad-hoc task.

3.5 Justifications

An important amount of work has been dedicated in the DL community to the computation of minimal subsets of an input KB which entail some (usually undesired) consequence(s). This problem is also closely related to the computation of maximal subsets which do not entail this consequence, as will be explained in Section 3.6.2.2.

Given a consequence ψ of an input KB K, a *justification* for ψ is a minimal subset of K having ψ as a consequence. Minimality is understood here wrt set inclusion, i.e. the set of all justifications for ψ in K is $\mathcal{J} = \min_{\subseteq}\{\Delta \subseteq K \mid \Delta \vdash \psi\}$. In other words, if $J \subseteq K$ is a justification for ψ, then $J \vdash \psi$, and for any $J' \subset J$, $J' \nvdash \psi$. A justification is a possible explanation for ψ being a consequence of K, which can be helpful for debugging K if ψ has been identified as an undesired consequence of K. There may be several justifications for a given $\psi \in \mathrm{Cn}(K)$ though, and they may overlap.

A case of particular interest is the one where $\psi = \top \sqsubseteq \bot$. In this case, a justification for ψ can be viewed as an explanation for the inconsistency of K, also called a *minimal conflict* in the diagnosis literature [Rei87]. Another well studied possibility for DLs is $\psi = A \sqsubseteq \bot$, where A is an atomic concept in the signature of K, i.e. $A \in N_{Con}(K)$. In this case, a justification for ψ has been called a MUPS (for minimal unsatisfiability preserving sub-TBox) in [SC03].

If ψ is equivalent to neither of these, and if K is consistent, then computing

justifications for ψ in K can generally be reduced to computing justifications for $\top \sqsubseteq \bot$ in $K \cup \{\overline{\psi}\}$, where $\overline{\psi}$ is a formula such that for all $\Delta \subseteq K$, $\Delta \cup \{\overline{\psi}\} \vdash \top \sqsubseteq \bot$ iff $\Delta \vdash \psi$. Then if J is a justification for $\top \sqsubseteq \bot$ in $K \cup \{\overline{\psi}\}$, it must be the case that $\overline{\psi} \in J$, and $J' = J \setminus \{\overline{\psi}\}$ is a justification for $\psi \in K$. As a simple example, if $\psi = C(e)$, with C a DL concept and e an individual, then $\overline{\psi} = \neg C(e)$ (up to equivalence). Another example is $\psi = C \sqsubseteq D$, with C and D DL concepts, and $\overline{\psi} = C \sqcap \neg D(e)$, with e a fresh individual, i.e. such that $e \notin N_{Ind}(K)$.

The notion of justification has also been generalized to a set Ψ of consequences. This is useful for most DLs, which are closed neither under conjunction nor disjunction. Two different generalizations can be made, depending on whether Ψ is understood conjunctively or disjunctively. In the conjunctive case, if $\Psi \subseteq \mathrm{Cn}(K)$, then a justification J for Ψ in K is a subset of K such that $\Psi \subseteq \mathrm{Cn}(J)$, and for all $J' \subset J$, $\Psi \not\subseteq \mathrm{Cn}(J')$. In the disjunctive case, if $\Psi \subseteq \mathrm{Cn}(K)$, then a justification J for Ψ in K is a subset of K such that $\mathrm{Cn}(J) \cap \Psi \neq \emptyset$, and for all $J' \subset J$, $\mathrm{Cn}(J) \cap \Psi = \emptyset$. In particular, in the disjunctive case, if $\Psi = \{A \sqsubseteq \bot \mid A \in N_{Con}(K)\}$, a justification for Ψ has been called a MIPS (for minimal incoherence preserving sub-TBox) in [SC03]. This last case corresponds to the notion of incoherence introduced in Chapter 2 Section 2.3.5, i.e. each MIPS is a justification for the incoherence of K, and conversely.

Laconic justification have also been defined and explored in depth in [Hor11]. They can be viewed as finer-grained form of explanations. Intuitively, a laconic justification J' is a weaker version of a justification J for ψ, such that each axiom of J has been individually weakened to produce J',[3] and such that $J' \vdash \psi$ still holds.

[3]It would not be perfectly accurate to say "maximally weakened" here, although this is clearly the general intuition. It turns out that the notion of laconic justification used in [Hor11] is more subtle. In particular, in some specific cases, there may be two laconic justifications J' and J'' for ψ, obtained out of a same (non-laconic) justification J, and such that $J'' \subset \mathrm{Cn}(J')$.

Among the motivations behind laconic justifications are the fact that J' is easier to read, but also that there may be several laconic justifications for a same J.

The rest of this section introduces some of the main algorithms proposed to compute (non-laconic) justifications for DLs, and is largely based on [Hor11]. References to these algorithms will be made in the following chapters, which is why the reviews provided here are more detailed than the reviews of the previous sections.

Two problems can be identified, computing one, and computing all justifications for a given (set of) consequence(s) of the input KB. A distinction introduced in [PSK05] is also traditionally made between so-called *glass-box*, and *black-box* algorithms. The former are modifications of some existing algorithm to check consistency of satisfiability, whereas the latter use a reasoner as an external library.

3.5.1 Glass-box algorithms

In $\mathcal{ALC}$ and more expressive DLs, a glass-box algorithm to compute justifications for a set of consequences of a KB is prototypically a customization of some existing tableau algorithm used to decide consistency or satisfiability, like the tableau for consistency briefly described in Chapter 2 Section 2.3.8.2. The technique was introduced in [BH95] to compute a so-called *pinpointing formula* (defined in section 3.5.1.2) for the inconsistency of a set Φ of statements. It was then specifically applied to the computation of one or several justifications by [SC03], [KPSH05] or [Lam07] among others, each work dealing with specific DLs and/or a particular set Ψ of undesired consequences (inconsistency, incoherence, ...). A more general formalization was also proposed by [BP10].

For the sake of readability, the focus is put here on the specific case where justifications for the inconsistency of a set Φ of axioms needs to be computed. But

as already mentioned at the beginning of Section 3.5, the generalization to an arbitrary set Ψ of undesired consequences is generally straightforward, because deciding entailment in these DLs can most often be reduced to deciding consistency.

3.5.1.1 Computing one justification.

As a reminder (see Chapter 2 Section 2.3.8.2 for details), if Φ is a (finite) set of DL axioms, a tableau algorithm to decide the consistency of Φ updates a family $\mathcal{W}$ of sets of DL formulas, whose cardinality may increase during the execution, and such that at each step, Φ is consistent iff there is a consistent $W \in \mathcal{W}$. Again, $\mathcal{W}_0, .., \mathcal{W}_n$ will designate the successive states of $\mathcal{W}$ during the execution. Let Γ be defined by $\Gamma = \{\top(e) \mid e \in N_{Ind}(\Phi)\}$ if $N_{Ind}(\Phi) \neq \emptyset$ and $\Gamma = \{\top(e')\}$ otherwise, with e' a fresh DL individual, and let $W_0 = \Phi \cup \Gamma$. A tableau algorithm for the consistency of Φ may start with $\mathcal{W}_0 = \{W_0\}$, provided a prior syntactic normalization of each axiom in Φ. The execution then consists in iteratively expanding each $W \in \mathcal{W}$, replacing W in $\mathcal{W}$ by one or two sets of statements, until either no expansion rule can be applied to some clash-free element of $\mathcal{W}$, or all of them contain a clash. A rule is triggered by the presence of some formulas $\{\gamma_1, .., \gamma_n\}$ in W, and produces one or two expansions of W. If W' is an expansion of W, then $W' = W \cup \Gamma'$, with Γ' a nonempty set of derived formulas. Again, expans$^*(W)$ will denote W and all expansions transitively derived from W during the execution.

In its simplest form, a glass-box algorithm to compute justifications proceeds by keeping track, for each derived DL formula γ in each $W \in \mathcal{W}$, of the axioms of Φ which were used to derive it. To this end, in each $W \in \mathcal{W}$, each $\gamma \in W$ is mapped to a subset lab$(W, \{\gamma\})$ of Φ.

For each $\phi \in \Phi$, lab$(W_0, \{\phi\})$ is initiated with lab$(W_0, \{\phi\}) = \{\phi\}$, and for each

$\gamma \in \Gamma$, $\text{lab}(W_0, \{\gamma\})$ is initiated with $\text{lab}(W_0, \{\gamma\}) = \emptyset$.

If $W^{'}$ is an expansion of W, then for each derived DL formula γ' in $W^{'} \setminus W$, the set of axioms $\text{lab}(W^{'}, \{\gamma'\})$ associated to γ' in $W^{'}$ is the union of the sets of axioms associated in W to each DL formula which triggered the rule. For instance, let us assume that $A(e) \in W$, $A \sqsubseteq C \in W$, but $C(e) \notin W$, such that W is expanded as $W^{'} = W \cup \{C(e)\}$. Then $\text{lab}(W^{'}, \{C(e)\}) = \text{lab}(W, \{A(e)\}) \cup \text{lab}(W, \{A \sqsubseteq C\})$. On the other hand, the sets of axioms assigned to non-derived formulas in $W^{'}$ (in this example, $A(e)$ and $A \sqsubseteq C$) remain unchanged, i.e. for each $\gamma \in W \cap W'$, $\text{lab}(W^{'}, \{\gamma\}) = \text{lab}(W, \{\gamma\})$.

As mentioned in Chapter 2 Section 2.3.8.2, if $\mathcal{W}$ at some point during the execution contains several sets of formulas, this is due the presence of disjunctions ($\sqcup$) in (the syntactically normalized version of) Φ, such that each $W \in \mathcal{W}$ can be viewed as an alternative attempt to build a model for Φ.

If Φ is inconsistent, then at the end of the execution, each $W \in \mathcal{W}_n$ contains a clash. Let $\text{clash}(W) \subseteq W$ denote the clash identified in some $W \in \mathcal{W}_n$. If $\text{lab}(W, \text{clash}(W))$ is the union the axioms of Φ associated to each $\gamma \in \text{clash}(W)$, i.e. if $\text{lab}(W, \text{clash}(W)) = \bigcup_{\gamma \in \text{clash}(W)} \text{lab}(W, \{\gamma\})$, then $\text{lab}(W, \text{clash}(W))$ can intuitively be viewed as an explanation for the inconsistency of W. Now if H is the union of all axioms involved in each clash after termination, i.e. if $H = \bigcup_{W \in \mathcal{W}} \text{lab}(W, \text{clash}(W))$, then H is an inconsistent subset of Φ, and as such can be viewed as an explanation for the inconsistency of Φ.

This inconsistent subset of Φ is not always minimal though. Here is a very simple example, borrowed from [Hor11].

Ex 3.5.1. $\phi_1 = A_1(e_1)$,
$\phi_2 = A_1 \sqsubseteq A_2 \sqcap \exists R.(A_3 \sqcap \neg A_3)$,

$\phi_3 = A_2 \sqsubseteq \bot$,
$\Phi = \{\phi_1, \phi_2, \phi_3\}$

If $\mathcal{W}_0$ is initiated with $\{W_0\}$ such that $W_0 = \Phi \cup \{\top(e_1)\}$, then $\text{lab}(W_0, \{\phi_1\}) = \{\phi_1\}$, $\text{lab}(W_0, \{\phi_2\}) = \{\phi_2\}$, $\text{lab}(W_0, \{\phi_3\}) = \{\phi_3\}$, and $\text{lab}(W_0, \{\top(e_1)\}) = \emptyset$. A first application of an expansion rule may yield $\mathcal{W}_1 = \{W_1\}$, with $W_1 = W_0 \cup \{A_2 \sqcap \exists R.(A_3 \sqcap \neg A_3)(e_1)\}$, and $\text{lab}(W_1, \{A_2 \sqcap \exists R.(A_3 \sqcap \neg A_3)(e_1)\}) = \{\phi_1, \phi_2\}$. Then expanding W_1 may yield $\mathcal{W}_2 = \{W_2\}$, with $W_2 = W_1 \cup \{A_2(e_1), \exists R.(A_3 \sqcap \neg A_3)(e_1)\}$, and for each $\gamma \in W_2 \setminus W_1$, $\text{lab}(W_2, \{\gamma\}) = \{\phi_1, \phi_2\}$. At this point, one may choose to extend W_2 either with $\{\bot(e_1)\}$, or with $\{R(e_1, e_2), A_3 \sqcap \neg A_3(e_2), \top(e_2)\}$. In the first case, this yields $\mathcal{W}_3 = \{W_3\}$, with $W_3 = W_2 \cup \{\bot(e_1)\}$, and $\text{lab}(W_3, \{\bot(e_1)\}) = \text{lab}(W_2, \{A_2(e_1)\}) \cup \text{lab}(W_2, \{\phi_3\}) = \Phi$. Because $\{\bot(e_1)\}$ is a clash in W_3 and $\mathcal{W}_3 = \{W_3\}$, the procedure terminates, and the explanation for the inconsistency is $H_1 = \bigcup_{W \in \mathcal{W}_3} \text{lab}(W, \text{clash}(W)) = \Phi$. In the second case, this yields $\mathcal{W}_3 = \{W_4\}$, with $W_4 = W_2 \cup \{R(e_1, e_2), A_3 \sqcap \neg A_3(e_2), \top(e_2)\}$, and for each $\gamma \in W_4 \setminus W_2$, $\text{lab}(W_4, \{\gamma\}) = \{\phi_1, \phi_2\}$. Finally, expanding W_4 one step further may yield $\mathcal{W}_4 = \{W_5\}$, with $W_5 = W_4 \cup \{A_3(e_2), \neg A_3(e_2)\}$, and for each $\gamma \in W_5 \setminus W_4$, $\text{lab}(W_5, \{\gamma\}) = \{\phi_1, \phi_2\}$. Similarly, because $\{A_3(e_2), \neg A_3(e_2)\}$ is a clash in W_5 and $\mathcal{W}_4 = \{W_5\}$, the procedure terminates. But in this case, the explanation for the inconsistency is $H_2 = \bigcup_{W \in \mathcal{W}_4} \text{lab}(W, \text{clash}(W)) = \{\phi_1, \phi_2\}$. So as $H_2 \subset H_1$, the first possible execution yields an inconsistent subset of Φ which is not a justification.

The solution to this problem proposed by [Hor11] consists in adding a subsequent black-box reduction phase, as described in Section 3.5.2, in order to obtain a minimal explanation for the inconsistency of Φ.

3.5.1.2 Computing all justifications

Glass-box algorithms used to compute all justifications are more complex customizations of tableau algorithms. The first reason is the termination condition of a standard tableau algorithm. Only one clash needs to be found in each $W \in \mathcal{W}_n$, i.e. if W contains a clash, it is not expanded any further. But if instead expansion rules can still be applied to an element of $\mathcal{W}$ which already contains a clash, then multiple clashes may be present in a same element of $\mathcal{W}_n$ (i.e. after termination). Example 3.5.1 above is an illustration, where the explanation for of the two clashes in W is a subset of the explanation for the other. But it may also be the case for some element $W \in \mathcal{W}_n$ that no explanation for a clashes in W is a subset of the explanations for all others. As a very simple illustration, let us consider the following input set of axioms Φ:

Ex 3.5.2. $\phi_1 = A_1(e)$,
$\phi_2 = \neg A_1(e)$,
$\phi_3 = \neg A_2(e)$,
$\phi_4 = A_1 \sqsubseteq A_2$,
$\Phi = \{\phi_1, \phi_2, \phi_3, \phi_4\}$

And let us assume that $\mathcal{W}_0$ is initiated with $\mathcal{W}_0 = \{W_0\}$, such that $W_0 = \Phi \cup \{\top(e)\}$. Then $\{\phi_1, \phi_2\}$ is the only clash in W_0. Because W_0 contains a clash and $\mathcal{W}_0 = \{W_0\}$, a standard tableau algorithm would immediately terminate with $\mathcal{W}_0 = \mathcal{W}_n = \{W_0\}$, yielding the explanation $H_1 = \bigcup_{W \in \mathcal{W}_n} \text{lab}(W, \text{clash}(W)) = \{\phi_1, \phi_2\}$. But expanding W_0 one step further may yield $\mathcal{W}_1 = \{W_1\}$, with $W_1 = W_0 \cup \{A_2(e)\}$, and $\text{lab}(W_1, \{A_2(e)\}) = \text{lab}(W_0, \{\phi_1\} \cup \text{lab}(W_0, \{\phi_4\}) = \{\phi_1, \phi_4\}$. Then W_1 contains two clashes, namely $\{\phi_1, \phi_2\}$ and $\{\phi_3, A_2(e)\}$. If the latter is selected as a clash in W_1 instead of the former, a different explanation H_2 will be obtained, namely $H_2 =$

$\{\phi_1, \phi_3, \phi_4\}$. And as opposed to example 3.5.1, H_1 and H_2 are both justifications for the inconsistency of Φ.

For this reason, the modified tableau algorithm proposed for instance in [SC03] is a *saturated tableau*, which means that expansion rules are applied to each $W \in \mathcal{W}$ even if W contains a clash, until no rule can be triggered anymore. After termination, each $W \in \mathcal{W}_n$ may contain several clashes, therefore clash(W) for a saturated tableau will designate in what follows a set of clashes, and not a single clash. Then each selection of one clash per $W \in \mathcal{W}_n$ may yield a different explanation, such that the number of computed explanations in the worst case is exponential in $|\mathcal{W}_n|$.

Another difference from the previous algorithm is due to the fact that if $W' \in$ expans$^*(W)$ for some $W \in \mathcal{W}_i$ and some $i \in \{1, .., n\}$, then there may be other alternative ways to derive W' from W, involving different subsets of Φ. Here is a simple example:

Ex 3.5.3. $\phi_1 = A_1(e_1)$,
$\phi_2 = A_1 \sqsubseteq A_2$,
$\phi_3 = \neg A_2(e_1)$,
$\phi_4 = R(e_2, e_1)$,
$\phi_5 = \top \sqsubseteq \forall R.A_2$,
$\Phi = \{\phi_1, \phi_2, \phi_3, \phi_4, \phi_5\}$

Let $\mathcal{W}_0 = \{W_0\}$, with $W_0 = \Phi \cup \{\top(e_1), \top(e_2)\}$. Then $W_1 = W_0 \cup \{A_2(e_1)\}$ is an expansion of W_0, and the derivation of W_1 can be triggered either by $\{\phi_1, \phi_2\}$, or by $\{\phi_4, \phi_5\}$. In addition, there is only one clash in any element of expans$^*(W_1)$, namely $\{A_2(e_1), \neg A_2(e_1)\}$, and in the absence of disjunction, even if expansion rules are applied until saturation, the cardinality of $\mathcal{W}$ after termination is still 1, i.e. $\mathcal{W}_n = \{W_m\}$ for some $W_m \in$ expans$^*(W_1)$. So the only explanation for the inconsistency of

Φ which can be computed during the execution is $\text{lab}(W_m, \{A_2(e_1), \neg A_2(e_1)\})$. But if the derivation of W_1 is triggered by $\{\phi_1, \phi_2\}$, then $\text{lab}(W_m, \{A_2(e_1), \neg A_2(e_1)\}) = \{\phi_1, \phi_2, \phi_3\}$, whereas if it is triggered by $\{\phi_4, \phi_5\}$, then $\text{lab}(W_m, \{A_2(e_1), \neg A_2(e_1)\}) = \{\phi_3, \phi_4, \phi_5\}$. In both cases, $\text{lab}(W_m, \text{clash}(W_m))$ is a justification, but depending on the order in which expansion rules are triggered, only one of them can be computed.

A solution to this problem consists in setting each element W of $\mathcal{W}$ to be a set of pairs $< \gamma, \Lambda >$ instead of a set of formulas, with $\Lambda \subseteq \Phi$. A clash in W in this case is a set of pairs $\{< \gamma_1, \Lambda_1 >, .., < \gamma_m, \Lambda_m >\}$. For instance, $\{< A(e), \Lambda_1 >, < \neg A(e), \Lambda_2 >\}$ is a possible clash. If M is a clash, let $r(M)$ designate the set of all axioms of Φ involved in M, i.e. $r(M) = \bigcup_{<\gamma_j, \Lambda_j> \in M} \Lambda_j$. And once again, let $\text{clash}(W)$ designate the set of all clashes present in some $W \in \mathcal{W}$.

Then saturated tableaux with tracing to compute $\mathcal{J}$ like the one implemented by [SC03], are based on the following observation:

Theorem 3.5.1.1. $\mathcal{J} = \min_{\subseteq}\{ \bigcup_{k=1}^{m} r(M_k) \mid < M_1, .., M_m > \in \prod_{W_k \in \mathcal{W}_n} \text{clash}(W_k)\}$.

In other words, for each selection $< M_1, .., M_m >$ of one clash per element W_k of $\mathcal{W}_n$, the union $H = \bigcup_{j=1}^{m} r(M_j)$ of the axioms of Φ involved in each of these clashes is an inconsistent subset of Φ. Then if $\mathcal{H}$ is the family of all these inconsistent subsets of Φ, the procedure guarantees that $\mathcal{J} \subseteq \mathcal{H}$, and because the elements of $\mathcal{J}$ are minimal wrt set inclusion, $\mathcal{J} = \min_{\subseteq} \mathcal{H}$ must hold. A detailed execution of such a tableau will be provided in Chapter 8 Section 8.3.2.

An alternative notation is used in [BP10], associating a boolean formula $\text{bool}(W, \gamma)$ to each $\gamma \in W$ for each $W \in \mathcal{W}$, provided a bijective mapping ρ from Φ to a set of propositional variables. For instance, if $\mathcal{W}_0$ is initiated with $\{W_0\}$, then $\text{bool}(W_0, \phi) = \rho(\phi)$ is used to label each $\phi \in \Phi \cap W_0$. The elements of some $W \in \mathcal{W}$ in this case are just sets of DL formulas, and not pairs as previously, and $\text{clash}(W)$

is just a family of sets of formulas. An expansion rule may be triggered even if the derived formula γ is already in W, but the boolean formula $\text{bool}(W, \gamma)$ in this case is updated, in order to account for the fact that γ may be derived from different sets of axioms. Formally, Let W' be the immediate expansion of some $W \in \mathcal{W}_i$ for some $i \in \{0, .., n\}$, triggered by $\{\gamma_1, .., \gamma_m\} \subseteq W$, and let $\text{bool}(W, \gamma) = \bot$ if $\gamma \notin W$. Then for each $\gamma' \in W'$, $\text{bool}(W', \gamma') = \text{bool}(W, \gamma') \vee (\bigwedge_{j \in \{1,..,m\}} \text{bool}(W, \gamma_j))$. For instance, if $\text{bool}(W', \gamma') = (\rho(\phi_2) \wedge \rho(\phi_6)) \vee (\rho(\phi_3) \wedge \rho(\phi_4))$, this would indicate that γ' has been derived independently from axioms ϕ_2 and ϕ_6 on the one hand, and axioms ϕ_3 and ϕ_4 on the other hand.

After saturation of the tableau algorithm, a *pinpointing boolean formula* $\alpha(\mathcal{W}_n)$ for $\mathcal{W}_n$ can be computed out of all boolean formulas for all $W \in \mathcal{W}_n$ and all $\gamma \in W$. It is defined (up to equivalence) by $\alpha(\mathcal{W}_n) = \bigwedge_{W \in \mathcal{W}_n} \bigvee_{H \in \text{clash}(W)} \bigwedge_{\gamma \in H} \text{bool}(W, \gamma)$. Then a subset J of Φ is a justification for the inconsistency of Φ iff $\{\rho(\phi) \mid \phi \in J\}$ is a prime implicant of $\alpha(\mathcal{W}_n)$, or equivalently, $J \in \mathcal{J}$ iff $\{\rho(\phi) \mid \phi \in J\}$ is a minimal valuation verifying $\alpha(\mathcal{W}_n)$.

3.5.2 Black-box algorithms

Black-box algorithms rely on a reasoner for DLs as an external program to check entailment, in order to find minimal subsets of an input KB K which have a given consequence ψ.

3.5.2.1 Computing one justification

Different methods have been devised in order to reduce the number of black-box consistency checks required to obtain a justification. Again, a recent detailed state of the art can be found in [Hor11].

Two phases are generally distinguished: expansion and reduction. The expansion phase consists in extending an initial subset Δ of the input KB K, until $\Delta \vdash \psi$. Optimizations for the expansion phase are prototypically based on the signatures of Δ and ψ, and/or on the prior computation of a module for $\text{sig}(\psi)$ in K. (n introduction to modularization in DLs can be found in [SSZ09]. Intuitively, given a signature S and a set of formulas Φ, a module for S in Φ is a subset Φ' of Φ such that for any $\psi \in \text{Cn}(\Phi)$ with $\text{sig}(\psi) \subseteq S$, $\Phi' \vdash \psi$ holds.

After the expansion phase, i.e. after a $\Delta \subseteq K$ has been found such that $\Delta \vdash \psi$, the reduction phase searches for a minimal subset of Δ verifying this condition. For any $\Delta' \subseteq \Delta$, if $\Delta' \vdash \phi$ holds, then any axiom $\phi \in \Delta \setminus \Delta'$ can be safely discarded. It may be the case that ϕ appears in a justification for ψ, but at least one justification is a subset of Δ', and the algorithm only searches for one of them. Optimizations for this phase include divide and conquer algorithms, used for instance in [FS05].

3.5.2.2 Computing all justifications.

As an alternative to a saturated tableau, [Hor11] uses a simplified version of Reiter's algorithm [Rei87] to compute all justifications for a given consequence ψ of K.

The algorithm makes repeated calls to an external procedure which computes one justification in different subsets of K implying ψ. The procedure which computes one justification can actually be a glass-box or a black-box one. So technically, in the former case, this approach may not be strictly considered as a black-box one, as some hybrid black/glass-box strategy.

Reiter's algorithm was initially proposed to compute diagnoses, and not justifications (although in practice it allows for the computation of both). This is why it is introduced in Section 3.6.2.2 of this chapter, dedicated to diagnosis. It is also

reproduced and discussed in more details in Chapter 8 Section 8.3.1.

3.6 Belief change

An important number of KB debugging techniques pertain more or less directly to the field of belief change (for an introduction, see [Gä03]). Belief change prototypically deals with weakening an input theory K in order either to forget some consequence ψ of K, or to accommodate for some additional knowledge θ (which, in the non-trivial case, contradicts K). The first operation is called a contraction, and the second is called either a revision or an update, depending on the requirements put on it.

At a very high level of description, two non-trivial constraints must be verified by such operations: no arbitrary choice should be made (i.e. the weakening of K must be deterministic), and no information should be unnecessarily lost in the process. This last informal constraint can be understood in different ways though, shaping the field.

Two very different views on the problem will be distinguished. According to the first one, a KB is primarily considered as a (deductively closed) theory, sometimes called a *belief set*. Constraints are then put on the deductive closures of the input and output KBs, regardless of their syntactic formulations.

The second view has been called *belief base* revision/contraction in [RW09], or *syntactic contraction/revision* in [GKZ12]. According to this view, the weakened base is required to be a syntactic subset of the input KB, i.e. weakening consists in discarding axioms of the input KB (although a few exceptions to this rule will also be presented). Debugging strategies corresponding to the second view are not always perceived as rightfully pertaining to belief change. Many of them have been presented as pertaining to knowledge engineering or diagnosis instead, without any explicit

reference to belief change. On the other hand, very similar works by [QHH+08] or [RW09] for example have been explicitly related to the literature on belief change. No claim is made here wrt to this terminological issue, and this classification is adopted for the sake of clarity only. The former family of approaches will be designated as *belief set contraction/revision*, and the latter as *syntax-based contraction/revision*.

3.6.1 Belief set contraction/revision

Two types of approaches to belief set contraction in DLs are distinguished in [CKNZ10, GKZ12], although they are not necessarily exclusive. The first view is called "formula-based", and is largely influenced by the AGM (for Atkinson, Gärdenfors and Makinson) paradigm for belief change. The second view is called "model-based", and primarily characterizes minimal information loss in terms of some ordering over interpretations.

3.6.1.1 Formula-based contraction/revision

Contraction and revision are defined in the AGM framework [AGM85] as binary operations, taking as input a (deductively closed) theory K and a formula, denoted here by ψ for contraction, and θ for revision. The original framework, primarily developed for propositional logic, will be briefly introduced, followed by its generalization to a wider class of logics, including many of the DLs of interest here.

Regardless of its syntactic formulation, the input knowledge K is viewed (from an abstract perspective) as a theory, i.e. $K = \mathrm{Cn}(K)$. Note that this convention is *not* in line with the one adopted in the other sections of this chapter, where K designates a KB, i.e. a finite set of axioms. The choice is made to follow (in the current section only) the more concise notation conventionally adopted in the belief

change literature.[4]

Given a theory K and a formula ψ, the contraction of K by ψ, noted $K - \psi$, consists in weakening K so that $K - \psi \nvdash \psi$.

Given a theory K and a formula θ the revision of K by θ, noted $K * \theta$, consists in modifying K such that $K*\theta \vdash \theta$, but $K*\theta \nvdash \bot$. Revision seems of a particular interest as a follow-up to the strategy discussed in chapter 7, which consists in extending a consistent but loosely constrained KB in order to yield an inconsistency, with some external ontological knowledge θ to be preserved.

Finally, the expansion of K by ϕ, noted $K + \phi$, is the deductive closure of $K \cup \phi$. Both $K - \psi$ and $K * \theta$ are also considered as deductively closed.

Contraction and revision are characterized by so-called *rationality postulates*, some of which are relatively obvious, dealing mostly with limit cases, and others being more controversial.

3.6.1.1.1 Contraction. The first five postulates for contraction are relatively simple.[5] The first one requires the output to be deductively closed, and the four other ones correspond to basic intuitions:

(1)	$K - \psi = \mathrm{Cn}(K - \psi)$	(closure)
(2)	if $\psi \notin \mathrm{Cn}(\emptyset)$, then $\psi \notin K - \psi$	(success)
(3)	$K - \psi \subseteq K$	(inclusion)
(4)	if $\psi \notin K$, then $K - \psi = K$	(vacuity)
(5)	if $\mathrm{Cn}(\{\psi_1\}) = \mathrm{Cn}(\{\psi_2\})$, then $K - \psi_1 = K - \psi_2$	(extensionality)

[4] Alternatively, it is also possible to consider K as a KB as previously, by replacing all occurrences of "K" in the rest of the current section by "Cn(K)".

[5] The conventional indices of the extensionality and recovery postulates have been swapped for the sake of the presentation: usually, the index of recovery is 5, whereas the index of extensionality is 6. In this more traditional setting, the above sentence becomes "The first four postulates for contraction together with the sixth one are relatively simple".

A contraction operator satisfying these five postulates is very loosely constrained. For instance, a contraction operator which returns K if $\psi \notin K$ and $\mathrm{Cn}(\emptyset)$ otherwise satisfies all five of them.

More controversial is the recovery postulate, which, together with the five others, guarantees a form of minimal information loss.

(6) $K \subseteq (K - \psi) + \psi$ (recovery)

Intuitively, after a contraction of K by ψ, adding ψ again to the output and closing it deductively should allow for the recovery of the whole initial knowledge K.

Alternatively, AGM contraction can be characterized in more constructive terms. A *remainder* for K and ψ is a maximal subset of K not entailing ψ, and the *remainder set* $K \perp \psi$ is the family of all remainders for K and ψ. It was shown in [AGM85] that an operator $-$ satisfies postulates 1 to 6 iff for any K and ψ such that ψ is not a tautology, it selects the intersection of more than one and less than all elements of $K \perp \psi$ (if ψ is a tautology, it simply returns K). This does not immediately provide an algorithm to compute $K - \psi$, but this intuitive construction, called *partial meet contraction*, legitimates in a way the postulates. [AGM85] show that in the (only) non-trivial case where $\psi \in K$ and ψ is not a tautology, selecting only one remainder would result in a belief set K' with the improbable property that $\mathrm{Cn}(K' \cup \{\neg\psi\})$ is a complete theory, i.e. that for any formula ψ', either $\psi' \in \mathrm{Cn}(K' \cup \{\neg\psi\})$ or $\neg\psi' \in \mathrm{Cn}(K' \cup \{\neg\psi\})$. Selecting all remainders and taking their intersection on the other hand would yield a KB K' with the other improbable property that $\mathrm{Cn}(K' \cup \{\neg\psi\}) = \mathrm{Cn}(\{\neg\psi\})$.

Two additional postulates, called the supplementary Gärdenfors postulates for contraction, are also used in the AGM framework, and may be more difficult to understand. They ensure that the behavior of an operator for a contraction by a

conjunction of two formulas remains in line with its behavior for a contraction by each conjunct.

(7) if $\psi_1 \notin K - (\psi_1 \wedge \psi_2)$, then $K - (\psi_1 \wedge \psi_2) \subseteq K - \psi_1$

(8) $(K - \psi_1) \cap (K - \psi_2) \subseteq K - (\psi_1 \wedge \psi_2)$

The first one guarantees that if one needs to forget ψ_1 in order to contract K by either ψ_1 or ψ_2 (i.e. by $\psi_1 \wedge \psi_2$), then nothing should be preserved when contracting by $\psi_1 \wedge \psi_2$ which is not preserved when contracting by ψ_1 alone. This is not guaranteed by the 6 first postulates. For instance, in propositional logic, let $K = \mathrm{Cn}(\{p, q\})$, and let $-$ be a contraction operator such that $K - p = \mathrm{Cn}(\{q \vee \neg p\})$ and $K - (p \wedge q) = \mathrm{Cn}(\{q\})$. Then both operations satisfy all 6 first postulates. In this case, a contraction by p or a contraction by $p \wedge q$ both result in the loss of p. But because $K - p \subset K - (p \wedge q)$, the latter operation is more advantageous, in that less information is lost. Therefore if one just wants to forget p, it is preferable to contract K by $p \wedge q$ rather than simply by p, which is arguably counterintuitive, or at least very impractical.

The second additional postulate guarantees that what is preserved when contracting both by ψ_1 and by ψ_2 is also preserved when contracting either by ψ_1 or by ψ_2. For instance, in propositional logic, let $K = \mathrm{Cn}(\{p, q\})$, and let $-$ be a contraction operator such that $K - p = \mathrm{Cn}(\{q\})$, $K - q = \mathrm{Cn}(\{p\})$, and $K - (p \wedge q) = \mathrm{Cn}(\emptyset)$. Then again, the operations $K - p$, $K - q$ and $K - (p \wedge q)$ satisfy all 6 first postulates for contraction. Let us assume that one wants to forget $p \wedge q$, i.e. to forget either p or q (or both). Then contracting both by p and by q (and taking their intersection) seems unnecessary (contracting by one of them is enough), and therefore should not be strictly preferable. But in this example, $K - (p \wedge q) = \mathrm{Cn}(\emptyset) \subset \mathrm{Cn}(\{p \vee q\}) = (K - p) \cap (K - q)$.

An important remark can be made here, which is that if postulate 6 (recovery) is dropped, satisfying postulates 7 and 8 together with the five first ones is not sufficient to guarantee minimal information loss. As a trivial example, postulates 1-5 and 7-8 are verified by the operator which for any K and ψ returns K if $\psi \notin K$, and $\mathrm{Cn}(\emptyset)$ otherwise.

Contraction operators satisfying postulates 1 to 8 have been shown in [AGM85] to be equivalent to constructions called *transitively relational partial meet contractions*, which will not be detailed here, because they are not essential to this work.

The generalization of AGM contraction to a larger class of logics has been investigated (among others) by [Flo06] and [Rib13], and a shorter version of some of their main contributions can be found in [RWFA13]. For DLs, a first generalization of this postulates addresses the problem of contracting K by a set Ψ of formulas, and not a single formula ψ. In this case, Ψ was interpreted conjunctively by [Flo06] and [Rib13]. The generalization of the five first postulates is then relatively straightforward. For instance, the success postulate becomes if $\mathrm{Cn}(\Psi) \neq \mathrm{Cn}(\emptyset)$, then $\Psi \not\subseteq K - \Psi$.

More importantly, the authors identified a series of sufficient and/or necessary conditions for a Tarskian logic to be *AGM compliant*, i.e. to admit a contraction operator which satisfies postulates 1 to 6 for any K and Ψ. It turns out that some of the most commonly used DLs are not AGM compliant. [Flo06] showed in particular that in many DLs (including $\mathcal{SHOIN}$, which underlies the OWL DL language), there are some K and Ψ such that no candidate subset of K for $K - \Psi$ can be found which satisfies these 6 postulates. The following counterexample is reproduced from [Flo06]. Let $K = \mathrm{Cn}(\{R \sqsubseteq S\})$, with R and S two DL atomic roles. Then $K \not\subseteq \mathrm{Cn}(K \setminus \{R \sqsubseteq S\})$. This is shown in [Flo06] with the following first-order interpretation $\mathcal{I} =< \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$ over $\mathrm{sig}(K)$, which is not a model of $\{R \sqsubseteq S\}$, but verifies the set K' of all strict consequences of K in $\mathcal{SHOIN}$.

Ex 3.6.1.

$\Delta^{\mathcal{I}} = \{a_1, a_2, b_1, b_2\}$
$R^{\mathcal{I}} = \{(a_1, b_1), (a_2, b_2), (b_1, a_1), (b_2, a_2)\}$
$S^{\mathcal{I}} = \{(a_1, b_2), (a_2, b_1), (b_2, a_1), (b_1, a_2)\}$

The existence of such an interpretation implies that $K' = \mathrm{Cn}(K') \subset K$. Let ψ be any strict and non-tautological consequence of K, i.e. $K \vdash \psi$ and $\psi \notin R \sqsubseteq S$, or equivalently $\psi \in K'$. Then let us assume that $-$ is a contraction operator in $\mathcal{SHOIN}$ satisfying postulates 1 to 5. By inclusion, $K - \psi \subseteq K$. Then because ψ is not-tautological, by success, $\psi \notin K - \psi$, and because $\psi \in \mathrm{Cn}(K)$, $K - \psi \subset K$, i.e. $K - \psi \subseteq K'$. Finally, because $\{\psi\} \subseteq K'$ as well, their union $\{\psi\} \cup (K - \psi) \subseteq K'$, and by monotonicity $\mathrm{Cn}(\{\psi\} \cup (K - \psi)) \subseteq \mathrm{Cn}(K') \subset K$, such that recovery is not satisfied by $-$.

As an alternative to recovery, [Rib13] proposed the *relevance* postulate, which was initially introduced by [Han91]:

> If $\gamma \in K \setminus (K - \psi)$, then there is a K' such that $K - \psi \subseteq K' \subseteq K$,
> $K' \not\vdash \psi$, but $K' \cup \{\gamma\} \vdash \psi$ (relevance)

Intuitively, if some formula γ is dropped from K (as a theory), then it must be involved in the derivation of ψ.

[Rib13] gives two arguments in favor of the relevance postulate in non AGM compliant logics. The first one is that in presence of the five first postulates, relevance can be shown to be equivalent to recovery in AGM compliant logics, such that relevance is in a way a generalization of recovery to a larger class of logics. The other argument is that for this larger class of logics, the five first postulates and relevance characterize partial meet revision just like the 5 first postulates and recovery for AGM compliant logics.

Similarly to the definition of an AGM compliant logic, the authors defined a *relevance compliant* logic as a logic for which a contraction operator $-$ can be defined, which for any K and ψ returns a belief set satisfying postulates 1 to 5 and relevance. They also identified different combinations of properties which may or may not be satisfied by a Tarskian logic (compactness, distributivity, or *decomposability* as defined in [Flo06], ...), and are sufficient and/or necessary conditions for a Tarskian logic to be either AGM compliant or relevance compliant.

3.6.1.1.2 Revision. Revision traditionally deals with the extension of a KB K with some formula θ. The prototypical application case is the one where θ is considered more reliable than K, and therefore if $K \cup \theta$ is inconsistent, only K should be weakened in order to accommodate for θ.

In propositional logic, revision and contraction have also been defined in terms of each other, by the two following equalities:

Definition 3.6.1.1. Correspondence between AGM contraction and revision

$$K * \theta = (K - \neg\theta) + \theta \qquad \text{(Levi identity)}$$
$$K - \psi = (K * \neg\psi) \cap K \qquad \text{(Harper identity)}$$

Most AGM postulates for revision correspond to a postulate for contraction, with the exception of consistency, which is relatively straightforward. Here are the 6 first AGM postulates for revision, also called the *basic* postulates for revision:

(1) $K * \theta = \mathrm{Cn}(K * \theta)$ (closure)
(2) $\theta \in K * \theta$ (success)
(3) $K * \theta \subseteq K + \theta$ (inclusion)
(4) if $\neg\theta \notin K$, then $K + \theta \subseteq K * \theta$ (vacuity)

(5) if $\{\theta\}$ is consistent, then $K * \theta$ is consistent (consistency)

(6) if $\mathrm{Cn}(\{\theta_1\}) = \mathrm{Cn}(\{\theta_2\})$, then $K * \theta_1 = K * \theta_2$ (extensionality)

An important difference with contraction though, discussed in depth in [Mak87], is the absence of a postulate for revision guaranteeing minimal change. This is important to understand the model-theoretic notions of minimal change described in the next section. Initially, a recovery postulate was proposed for revision, directly based on the above Harper identity:

$$K \subseteq ((K * \theta) \cap K) + \neg\theta \qquad \text{(recovery for revision)}$$

Intuitively, revising K by θ, keeping only the part of the output which was in K initially, and then extending it with $\neg\theta$ should allow for the recovery of K.

But although this may seem counterintuitive, the recovery postulate for revision is trivially satisfied in propositional logic by any revision operator which satisfies only the 6 first basic postulates, which can be easily shown (as done for instance by [Mak87]):

For the limit case where $\neg\theta \notin K$, from inclusion and vacuity, $K * \theta = K + \theta \supseteq K$, so $((K * \theta) \cap K) = K$, and $((K * \theta) \cap K) + \neg\theta = K + \neg\theta \supseteq K$. For the non-trivial case where $\neg\theta \in K$, take any $\psi \in K$. Then $\psi \vee \theta \in K$. By success, $\theta \in K * \theta$ also holds, so $\psi \vee \theta \in K * \theta$ as well, and therefore $\psi \vee \theta \in ((K * \theta) \cap K)$. Then $((K * \theta) \cap K) + \neg\theta$ contains both $\psi \vee \theta$ and $\neg\theta$, and because it is closed deductively, it contains ψ as well.

Like for contraction, two additional (and possibly less obvious) postulates, called the supplementary Gärdenfors postulates for revision, are commonly used:

(7) $K * (\theta_1 \wedge \theta_2) \subseteq (K * \theta_1) + \theta_2$

(8) if $\neg\theta_2 \notin K * \theta_1$, then $(K * \theta_1) + \theta_2 \subseteq K * (\theta_1 \wedge \theta_2)$

They prevent some odd behaviors of a revision operator wrt conjunctive formulas. To see this, in propositional logic, let $K = \mathrm{Cn}(\{\neg p, r\})$, and let $*$ be a revision operator such that $K * p = \mathrm{Cn}(\{p\})$ and $K * (p \wedge q) = \mathrm{Cn}(\{p, q, r\})$. These two operations satisfy all 6 basic postulates for revision. But learning p and then q results in a more important information loss that learning p and q altogether, because $(K * p) + q = \mathrm{Cn}(\{p, q\}) \subset K * (p \wedge q) = \mathrm{Cn}(\{p, q, r\})$, i.e. revising K by $p \wedge q$ is more interesting than revising K by p and then by q. Conversely, for postulate 8, let $K = \mathrm{Cn}(\{\neg p, r\})$, and let $*$ be a revision operator such that $K * p = \mathrm{Cn}(\{p, r\})$, and $K * (p \wedge q) = \mathrm{Cn}(\{p, q\})$. Then learning p and q altogether results in a more important information loss that learning p and then q, even though $(K * p) \cup \{q\}$ is consistent, because $K * (p \wedge q) = \mathrm{Cn}(\{p, q\}) \subset (K * p) + q = \mathrm{Cn}(\{p, q, r\})$.

It is important to note though that adding postulates 7 and 8 still does not guarantee minimal information loss for revision. In particular, postulates 1 to 8 are satisfied by the trivial revision operator $*$ defined by $K * \theta = \mathrm{Cn}(\{\theta\})$ if $K \cup \{\theta\}$ is inconsistent, and $K * \theta = \mathrm{Cn}(K \cup \{\theta\}) = K + \theta$ otherwise.

For AGM compliant logics, this legitimates the usage of an additional semantic notion of minimal change. [KM89] explicitly made the link between the AGM postulates for revision and several model-theoretic operators for belief change in propositional logics developed independently. They proved in particular the following result. A *faithful assignment* is defined as a function which to each satisfiable input belief set K associates a total preorder $\preceq_K$ (reflexive, transitive, but not necessarily antisymmetric) over the set M of all model which can be built over $\mathrm{sig}(K)$, and such that the models of K are all maximal wrt $\preceq_K$ in M, and no other model in M is. Let $*$ be a revision operator for propositional logics which satisfies all 6 basic postulates for revision. Then $*$ also satisfies postulates 7 and 8 iff there is a faithful assignment such that for any consistent K and consistent θ, $K * \theta$ is the belief set

whose models are maximal wrt $\preceq_K$ within the models of $\{\theta\}$.

Similarly to contraction, [Rib13] proposed a generalization of the first 6 AGM postulates to revision by a set Θ of formulas, and to a wider class of logics. Like for contraction, the generalization of these 6 postulates to a set of formulas (understood conjunctively) is relatively straightforward. The inclusion postulate for instance becomes $K * \Theta \subseteq \mathrm{Cn}(K \cup \Theta)$. But borrowing from [Del08], the postulates were also generalized in order to be applicable to logics where the negation of a formula cannot always be expressed, which is the case of most DLs.[6] In particular, the condition "if $\neg\theta \notin K$" in the vacuity postulate is replaced by "if $K \cup \Theta$ is consistent", and extensionality is replaced by the following *uniformity* postulate: if for all $K' \subseteq K$, $K' * \Theta_1$ is consistent iff $K' * \Theta_2$ is consistent, then $K \cap (K * \Theta_1) = K \cap (K * \Theta_2)$.

A minimal change postulate for revision for a larger class of logics was also introduced by [Rib13], which is an adaptation of the relevance postulate for contraction using Harper's identity, and from that regard is very similar to the abovementioned (redundant) recovery postulate for revision in AGM compliant logics:

> If $\gamma \in K \setminus (K * \Theta)$, then there is a K' such that $K \cap (K * \Theta) \subseteq K' \subseteq K$, $K' \cup \Theta$ is consistent, but $K' \cup \Theta \cup \{\gamma\}$ is inconsistent.
>
> (relevance for revision)

Intuitively, the belief in γ is lost when revising K by Θ only if γ is involved in the inconsistency of $K \cup \Theta$.[7] Finally, a construction comparable to partial meet was proposed, but allowing revision for logics which do not support full negation. The output of a revision process is required to be the intersection of some maximal subsets of K consistent with Θ, extended with Θ, and then closed deductively. The

[6] This is illustrated in Appendix A Section A.1.2.

[7] It is important to note that this is only a necessary condition to give up γ, not a sufficient one.

authors showed that relevance for revision, together with closure, success, inclusion, consistency and uniformity characterized this construction both for AGM compliant logics and for some non-compliant ones (namely compact and distributive Tarskian logics).

But Appendix A a property which, if verified by a Tarskian logic, guarantees that a revision operator in this logic which satisfies some of the basic postulates for revision (namely closure, success and vacuity) also satisfies relevance for revision. As a consequence, for Tarskian logics verifying this property, a trivial revision operator which returns $K+\Theta$ if $K \cup \Theta$ is consistent, and $\mathrm{Cn}(\Theta)$ otherwise, does satisfy all basic postulates for revision and relevance. In other words, for these logics, the relevance postulate for revision does not guarantee minimal information loss. Appendix A shows in particular that the DL $\mathcal{ALCHO}$ verifies this property, and therefore the following:

Proposition 3.6.1.1. If a revision operator in $\mathcal{ALCHO}$ satisfies closure, success, and vacuity, then it satisfies relevance.

$\mathcal{ALCHO}$ is the the extension of the DL $\mathcal{ALC}$ (introduced in Chapter 2 Section 2.3.2) with atomic role inclusions (i.e. formulas of the form $R \sqsubseteq S$) and nominals (i.e. concepts defined extensively, like $\{e_1, e_2\}$). $\mathcal{ALCHO}$ is an interesting case because it was identified by [Flo06] as one of the DLs for which recovery could not always be verified. This also raises the question of minimal change for revision in DLs which are more expressive than $\mathcal{ALCHO}$ and less expressive than $\mathcal{SROIQ}$.

The implications for $\mathcal{ALCHO}$ (and possibly more expressive DLs) are very similar to the implications of the redundancy of the recovery postulate for revision in AGM compliant logics, namely that some other principle of minimal change must be introduced for revision to be effective, which leads to the investigation of model-theoretic

notions of minimal change for DLs, also called *model-based* in the terminology of [CKNZ10, GKZ12].

3.6.1.2 Model-based contraction/revision

Model-based approaches of contraction or revision use some ordering over interpretations to characterize minimal information loss.

The output KB must satisfy basic intuitions about either contraction or revision, namely the above success and inclusion postulates for contraction, and success and consistency for revision. But in addition, the models of the output should be as close as possible to the models of the input KB K.[8] There are different ways to understand "as close as possible" though.

To our knowledge, most model-based contraction/revision approaches for DLs assume that the unique name assumption is made, i.e. that if S is a DL signature and if $\mathcal{I} =< \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$ is an interpretation over S, then for each pair of individuals $e_1, e_2 \in S$ such that $e_1 \neq e_2$, $e_1^{\mathcal{I}} \neq e_2^{\mathcal{I}}$ holds as well. This is not the semantic recommended by the W3C for OWL,[9] but this assumption will be made in this section in order to review these approaches accurately.

Additionally, it will be assumed that all first-order interpretations over the same signature S share the same (countably infinite) domain Δ, and that if $\mathcal{I}_1$ and $\mathcal{I}_2$ are two interpretations over S, then for each individual $e \in S$, $e^{\mathcal{I}_1} = e^{\mathcal{I}_2}$. Alternatively, [QLB06] uses the notion of a *preinterpretation* $\pi =< \Delta^{\pi}, .^{\pi} >$, which maps all individuals of S to an element of Δ^{π}. Then only interpretations over S which share

[8] As explained at the beginning of the previous section (Section 3.6.1.1), the notational convention $K = \mathrm{Cn}(K)$ was only adopted through Section 3.6.1.1, and does not hold anymore. So "K" in the current section (and the rest of this chapter) stands for a KB, i.e. a finite set of axioms, and its deductive closure is explicitly designated with "$\mathrm{Cn}(K)$".

[9] or OWL 2

the same preinterpretation can be compared. Adopting one or the other view has no incidence on the following observations, and therefore the former is preferred in order to simplify notation.

3.6.1.2.1 Typology. This section is partly based on the typology of model-based contraction techniques for DLs provided in [CKNZ10, GKZ12]. Most of them are inspired by existing techniques for propositional logics. The authors identify three (binary) distinctions which allow for a classification of the field, yielding in theory $2^3 = 8$ different approaches to the problem.

Set inclusion VS cardinality-based symmetric difference The first distinction will be introduced through propositional logic, in order to keep things simple. In propositional logics, an interpretation can be represented as the set of variables evaluated to true in it. Let $\mathcal{I}_0$ be an interpretations over a set P of propositional variables, and M a set of interpretations over P. A common way to obtain the interpretations in M which are the closest to $\mathcal{I}_0$ consists in using the symmetric difference between $\mathcal{I}_0$ and each $\mathcal{I} \in M$, i.e. $\text{diff}(\mathcal{I}_0, \mathcal{I}) = (\mathcal{I}_0 \setminus \mathcal{I}) \cup (\mathcal{I} \setminus \mathcal{I}_0)$. Then if $\mathcal{I}, \mathcal{I}' \in M$, $\mathcal{I}$ can be considered strictly closer to $\mathcal{I}_0$ than $\mathcal{I}'$ iff $|\,\text{diff}(\mathcal{I}_0, \mathcal{I})| < |\,\text{diff}(\mathcal{I}_0, \mathcal{I}')|$. This is the ordering of interpretations used for instance in [Dal88]. Alternatively, $\mathcal{I}$ can be considered strictly closer to $\mathcal{I}_0$ than $\mathcal{I}'$ iff $\text{diff}(\mathcal{I}_0, \mathcal{I}) \subset \text{diff}(\mathcal{I}_0, \mathcal{I}')$. In this latter case, "closer to $\mathcal{I}_0$" is not a total, but a partial preorder over M. This opposition between cardinality and set inclusion to order differences between interpretations is one of the three distinctions listed in [CKNZ10, GKZ12].

Atoms VS signature-based difference between interpretations The second distinction pertains to the representation of a first-order interpretation, which

is less straightforward than for a propositional interpretation (even for first-order interpretations without functions, which is the case for DLs). Let $\mathcal{I} < \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$ be an interpretation over a signature S. The *atom-based* view represents $\mathcal{I}$ as a set atoms($\mathcal{I}$) of formulas of the form $A(x)$ or $R(x_1, x_2)$, were A and R are respectively an atomic DL concept and an atomic DL role in S, and x, x_1 and x_2 are elements of $\Delta^{\mathcal{I}}$, such that $A(x)$ (resp. $R(x_1, x_2)$) $\in$ atoms($\mathcal{I}$) iff $x \in A^{\mathcal{I}}$ (resp. $(x_1, x_2) \in R^{\mathcal{I}}$). Then if $\mathcal{I}'$ is another interpretations over S, the difference between $\mathcal{I}$ and $\mathcal{I}'$ can be defined as above as the semantic difference between atoms($\mathcal{I}$) and atoms($\mathcal{I}'$), i.e. $\text{diff}(\mathcal{I}, \mathcal{I}') = (\text{atoms}(\mathcal{I}) \setminus \text{atoms}(\mathcal{I}')) \cup (\text{atoms}(\mathcal{I}') \setminus \text{atoms}(\mathcal{I}))$. Alternatively, the *signature-based* view represents the difference between $\mathcal{I}$ and $\mathcal{I}'$ as the set of atomic concepts and atomic roles in S which are interpreted differently by $\mathcal{I}$ and $\mathcal{I}'$, i.e. $\text{diff}(\mathcal{I}, \mathcal{I}') = \{X \in S \mid X^{\mathcal{I}} \neq X^{\mathcal{I}'}\}$.

Local VS global ordering of interpretations. The third distinction opposes *local* and *global* ordering of interpretations, and was notoriously made in [Men91] for propositional logic. For revision (but not for contraction), the operation based on *local* interpretations ordering has also been called belief *update*.

The local ordering of interpretations is defined as follows. For a contraction of Cn(K) by ψ, if $\mathcal{I}$ is a interpretation over sig(K) such that $\mathcal{I} \not\models \psi$, then $\mathcal{I} \in \text{mod}(\text{Cn}(K) - \psi)$ iff there is an $\mathcal{I}_0 \in \text{mod}(K)$ such that for any $\mathcal{I}'$ over sig(K) with $\mathcal{I}' \not\models \psi$, $\text{diff}(\mathcal{I}_0, \mathcal{I}) \leq \text{diff}(\mathcal{I}_0, \mathcal{I}')$ (or $\text{diff}(\mathcal{I}_0, \mathcal{I}) \subseteq \text{diff}(\mathcal{I}_0, \mathcal{I}')$, depending on whether cardinality or set inclusion is used to order differences between interpretations). Intuitively, the interpretations which do not verify ψ and are the closest to each model of K taken individually are retained. This is similar for revision. If $K \cup \{\theta\}$ is inconsistent, and if $\mathcal{I} \in \text{mod}(\{\theta\})$, then $\mathcal{I} \in \text{mod}(\text{Cn}(K) * \theta)$ iff there is an $\mathcal{I}_0 \in \text{mod}(K)$ such that for all $\mathcal{I}' \in \text{mod}(\{\theta\})$, $\text{diff}(\mathcal{I}_0, \mathcal{I}) \leq \text{diff}(\mathcal{I}_0, \mathcal{I}')$ (or $\text{diff}(\mathcal{I}_0, \mathcal{I}) \subseteq \text{diff}(\mathcal{I}_0, \mathcal{I}')$,

depending on whether cardinality or set inclusion is used to order differences between interpretations). In the case where $K \cup \theta$ is consistent, [Win88] proposed to use this operation as well, whereas [Bor85] proposed to comply to the vacuity postulate and return $\mathrm{Cn}(K) + \theta$.

Revision based on such local ordering of interpretations, also called belief update, typically deals with cases where some new information θ contradicts K by making it obsolete. For instance let us assume that $K \vdash \texttt{Planet}(\mathit{Pluto})$, and $\theta = \neg\texttt{Planet}(\mathit{Pluto})$. By default, the assumption is made here that K was accurate, in the sense that the models of K were among the "admissible worlds" intended by the knowledge engineer before the update. Or at least that some of them were, but in the absence of further information, and in virtue of the abovementioned determinism principle, all of them are assumed to be. Then the update operation minimally modifies each model of K in order to accommodate for θ, i.e. it produces as an output the theory $\mathrm{Cn}(K) * \theta$ whose models verify θ and are the closest to each model of K taken individually.

As noted by [KM89], all 8 postulates for revision in AGM compliant logics cannot be satisfied by an update operation when the difference between two interpretations is based on set inclusion. But it can actually be shown that this also holds when it is based on cardinality. Consider for instance the following example:

Ex 3.6.2. $K, \{\theta_1\}$ and $\{\theta_2\}$ are consistent.
$\mathrm{mod}(K) = \{\mathcal{I}_1, \mathcal{I}_2\}$.
$K \cup \{\theta_1\}$ is inconsistent, $K \cup \{\theta_2\}$ is inconsistent.
$\mathrm{mod}(\{\theta_1\}) = \{\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3\}$
$\mathrm{mod}(\{\theta_2\}) = \{\mathcal{J}_1, \mathcal{J}_3\}$
$|\,\mathrm{diff}(\mathcal{J}_1, \mathcal{I}_1)| = 1$, $|\,\mathrm{diff}(\mathcal{J}_1, \mathcal{I}_2)| = 3$

$|\operatorname{diff}(\mathcal{J}_2, \mathcal{I}_1)| = 1$, $|\operatorname{diff}(\mathcal{J}_2, \mathcal{I}_2)| = 1$

$|\operatorname{diff}(\mathcal{J}_3, \mathcal{I}_1)| = 2$, $|\operatorname{diff}(\mathcal{J}_3, \mathcal{I}_2)| = 2$

As already mentioned in Section 3.6.1.1.2, [KM89] showed that a revision operator satisfies all 8 postulates iff there is a faithful assignment which maps each consistent theory $\mathrm{Cn}(K)$ to a total preorder $\preceq_{\mathrm{Cn}(K)}$ over the interpretations of $\mathrm{sig}(K)$, such that for any consistent K and consistent $\{\theta\}$, $\mathrm{Cn}(K) * \theta$ is the belief set whose models are maximal wrt $\preceq_{\mathrm{Cn}(K)}$ among the models of $\{\theta\}$. Let $*$ be an update operator which relies on the cardinality of the symmetric difference between two interpretations (instead of set inclusion), and let us assume that $*$ satisfies all 8 postulates. In example 3.6.2, according to the above definition of an update operator, $\mathrm{mod}(\mathrm{Cn}(K) * \theta_1) = \{\mathcal{J}_1, \mathcal{J}_2\}$ must hold, because $\mathcal{J}_2$ is the model of $\{\theta_1\}$ closest to $\mathcal{I}_2$, and $\mathcal{J}_1$ and $\mathcal{J}_2$ are the two models of $\{\theta_1\}$ closest to $\mathcal{I}_1$. Therefore $\mathcal{J}_3 \in \mathrm{mod}(\{\theta_1)\} \setminus \mathrm{mod}(\mathrm{Cn}(K) * \theta_1)$, and so $\mathcal{J}_1 \prec_{\mathrm{Cn}(K)} \mathcal{J}_3$ should hold. But $\mathrm{mod}(\mathrm{Cn}(K) * \theta_2) = \{\mathcal{J}_1, \mathcal{J}_3\}$, because $\mathcal{J}_1$ is the model of $\{\theta_2\}$ closest to $\mathcal{I}_1$, and $\mathcal{J}_3$ is the model of $\{\theta_2\}$ closest to $\mathcal{I}_2$. Therefore $\mathcal{J}_1 \preceq_{\mathrm{Cn}(K)} \mathcal{J}_3$ and $\mathcal{J}_3 \preceq_{\mathrm{Cn}(K)} \mathcal{J}_1$ should hold, but the latter contradicts $\mathcal{J}_1 \prec_{\mathrm{Cn}(K)} \mathcal{J}_3$.

The global ordering of interpretations on the other hand is defined as follows. For contraction, if $\mathcal{I}$ is an interpretation over $\mathrm{sig}(K)$ with $\mathcal{I} \not\models \psi$, then $\mathcal{I} \in \mathrm{mod}(\mathrm{Cn}(K) - \psi)$ iff there is an $\mathcal{I}_0 \in \mathrm{mod}(K)$ such that for all $\mathcal{I}'$ over $\mathrm{sig}(K)$ with $\mathcal{I}' \not\models \psi$, and for all $\mathcal{I}'_0 \in \mathrm{mod}(K)$, $\operatorname{diff}(\mathcal{I}_0, \mathcal{I}) \leq \operatorname{diff}(\mathcal{I}'_0, \mathcal{I}')$ (or $\operatorname{diff}(\mathcal{I}_0, \mathcal{I}) \subseteq \operatorname{diff}(\mathcal{I}'_0, \mathcal{I}')$, depending on whether cardinality or set inclusion is used to order differences between interpretations). In other words, an interpretation $\mathcal{I}$ is retained iff it does not verify ψ and there is a model $\mathcal{I}_0$ of K such that $\operatorname{diff}(\mathcal{I}, \mathcal{I}_0)$ is minimal among all $\operatorname{diff}(\mathcal{I}', \mathcal{I}'_0)$ for all pairs $< \mathcal{I}', \mathcal{I}'_0 > \in \mathrm{mod}(\{\psi\}) \times \mathrm{mod}(K)$. For revision, if $\mathcal{I} \in \mathrm{mod}(\{\theta\})$, then $\mathcal{I} \in \mathrm{mod}(\mathrm{Cn}(K) * \theta)$ iff there is an $\mathcal{I}_0 \in \mathrm{mod}(K)$ such that for all $\mathcal{I}' \in \mathrm{mod}(\{\theta\})$,

and for all $\mathcal{I}'_0 \in \text{mod}(K)$, $\text{diff}(\mathcal{I}_0, \mathcal{I}) \leq \text{diff}(\mathcal{I}'_0, \mathcal{I}')$ (or $\text{diff}(\mathcal{I}_0, \mathcal{I}) \subseteq \text{diff}(\mathcal{I}'_0, \mathcal{I}')$, depending on whether cardinality or set inclusion is used to order differences between interpretations).

The intuition behind this last operation is that K was already incorrect before learning θ, as opposed to belief update. This could be the case for instance if $K \vdash \texttt{director}(\textit{Citizen Kane}, \textit{Woody Allen})$, and $\theta = \neg\texttt{director}(\textit{Citizen Kane}, \textit{Woody Allen})$ is learned. But also for the kind of errors more specifically targeted in this thesis, for instance if $K \vdash \texttt{Award}(\textit{Woddy Allen})$ and $\theta = \neg\texttt{Award}(\textit{Woody Allen})$, or if $K \vdash \texttt{Award} \sqcap \texttt{Person}(\textit{Woody Allen})$ and $\theta = \texttt{Award} \sqsubseteq \neg\texttt{Person}$. In all these cases, none of the models of K was correct before learning θ.

As opposed to update, all 8 postulates for revision can be satisfied in propositional logic by a revision operator based on a global ordering of interpretations, provided the difference between two interpretations is based on cardinality, as shown in [KM89]. But the following example shows that this does not hold in the case where the difference between two interpretations is based on set inclusion:

Ex 3.6.3. $K, \{\theta_1\}$ and $\{\theta_2\}$ are consistent.
$\text{mod}(K) = \{\mathcal{I}_1, \mathcal{I}_2\}$.
$K \cup \{\theta_1\}$ is inconsistent, $K \cup \{\theta_2\}$ is inconsistent.
$\text{mod}(\{\theta_1\}) = \{\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3\}$
$\text{mod}(\{\theta_2\}) = \{\mathcal{J}_1, \mathcal{J}_3\}$
$\text{diff}(\mathcal{J}_1, \mathcal{I}_1) = \{s\}$, $\text{diff}(\mathcal{J}_1, \mathcal{I}_2) = \{t\}$
$\text{diff}(\mathcal{J}_2, \mathcal{I}_1) = \{p\}$, $\text{diff}(\mathcal{J}_2, \mathcal{I}_2) = \{q\}$
$\text{diff}(\mathcal{J}_3, \mathcal{I}_1) = \{p, r\}$, $\text{diff}(\mathcal{J}_3, \mathcal{I}_2) = \{q, r\}$

The models of $\{\theta_1\}$ which are closest to any model of K are $\mathcal{J}_1$ and $\mathcal{J}_2$, so $\text{mod}(\text{Cn}(K) * \theta_1) = \{\mathcal{J}_1, \mathcal{J}_2\}$, and therefore $\mathcal{J}_3 \in \text{mod}(\{\theta_1)\} \setminus \text{mod}(\text{Cn}(K) * \theta_1)$, and

$\mathcal{J}_1 \prec_{\mathrm{Cn}(K)} \mathcal{J}_3$ should hold, similarly to example 3.6.2 above. But the models of $\{\theta_2\}$ which are closest to any model of K are $\mathcal{J}_1$ and $\mathcal{J}_3$, so $\mathrm{mod}(\mathrm{Cn}(K) * \theta_2) = \{\mathcal{J}_1, \mathcal{J}_3\}$, and therefore $\mathcal{J}_1 \preceq_{\mathrm{Cn}(K)} \mathcal{J}_3$ and $\mathcal{J}_3 \preceq_K \mathcal{J}_1$ must hold, which contradicts $\mathcal{J}_1 \prec_{\mathrm{Cn}(K)} \mathcal{J}_3$.

Some other proposals have been made for DLs which do not exactly fit into these categories. In particular [MLB05] and [QLB06] use a hybrid semantico-syntactic notion of minimal change for revision. If K is a DL KB (not necessarily closed deductively), and if $\mathcal{I} =< \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$ is an interpretation over $\mathrm{sig}(K)$, then $e(K, \mathcal{I})$ designates the cumulated number of "exceptions" to each axiom of K in $\mathcal{I}$. An exception to an axiom ϕ is understood here as an element x of $\Delta^{\mathcal{I}}$ such that $x \in (C \sqcap \neg D)^{\mathcal{I}}$ if ϕ is of the form $C \sqsubseteq D$, with C and D DL concepts, or such that $x = e^{\mathcal{I}}$ and $x \in (\neg C)^{\mathcal{I}}$ if $\phi = C(e)$, with C a DL concept and e an individual. Then for each KB K (not necessarily closed deductively), a total preorder $\preceq_K$ over the models of $\{\theta\}$ can be defined by $\mathcal{I} \preceq_{\mathrm{Cn}(K)} \mathcal{I}'$ iff $e(K, \mathcal{I}) \leq e(K, \mathcal{I}')$.

3.6.1.2.2 Inadequacy. Expressiveness issues for model-based contraction an revision were identified in two of the most common tractable DLs, namely *DL-Lite* and $\mathcal{EL}$, in [CKNZ10] (for revision in *DL-Lite*) and [GKZ12] (for contraction in *DL-Lite* and $\mathcal{EL}$).

For each combination of the 3 abovementioned criteria (difference between interpretations based on cardinality VS set inclusion, atom VS signature-based representation, and local VS global ordering of interpretations), there may be an optimal set of interpretations selected by a model-based operator, but such that no finite set of statements in these logics is verified exactly by these models. They also suspect similar inexpressibility issues for more expressive logics.

Another criticism of model-based belief contraction/revision for DLs pertains

to minimal information loss, in the case where the input KB's signature contains individuals. For *signature-based* representations of interpretations, this is relatively obvious. For instance, let $K = \{A(e_1), A(e_2), B(e_1)\}$, with A and B are atomic concepts, and e_1 and e_2 two individuals. And let us assume that $\mathrm{Cn}(K)$ should be contracted by $A(e_2)$ (or revised by $\neg A(e_2)$). Then any model $\mathcal{I}_0$ of K is such that $e_2^{\mathcal{I}_0} \in A^{\mathcal{I}_0}$, and any model $\mathcal{I}$ of the output theory must be such that $e_2^{\mathcal{I}} = e_2^{\mathcal{I}_0} \notin A^{\mathcal{I}}$,[10] so necessarily $A^{\mathcal{I}} \neq A^{\mathcal{I}_0}$. Now take the only interpretation $\mathcal{I}$ over $\mathrm{sig}(K)$ such that $A^{\mathcal{I}} = \emptyset$, and $B^{\mathcal{I}} = B^{\mathcal{I}_0}$. If a *signature-based* representations of interpretations is adopted, then the difference between $\mathcal{I}$ and any model of K is $\{A\}$, and it is minimal, no matter whether minimality is defined wrt cardinality or set inclusion. As a consequence, no matter whether a local or global ordering of interpretations is used, $\mathcal{I}$ is a model of the output KB, and therefore $A(e_1)$ is not a consequence of this output KB, which can be viewed as an unnecessary information loss.

For *atom-based* representations of interpretations, a similar observation can be made for $K = \{R(e_1, e_2)\}$, with R an atomic DL role and e_1 and e_2 two individuals, and a contraction by $R(e_1, e_2)$ (or a revision by $\neg\exists R.\{e_2\}(e_1)$). There must be a model $\mathcal{I}$ of the output KB such that $\mathcal{I} \not\models \exists R.\top(e_1)$, which once again can be viewed as an unnecessary information loss.

But more fundamentally, the model-based characterizations of minimal change proposed for DLs are not adapted to the problem of concern here due to their similarity with default reasoning, i.e. the fact that minimizing information loss according to these views often amounts to hard-coding exceptions. This will be illustrated with two examples, both of which are composed of real DBpedia [MJB12] statements. As

[10] As a reminder, it is assumed that all first-order interpretations over the same signature S share the same (countably infinite) domain Δ, and that if $\mathcal{I}_1$ and $\mathcal{I}_2$ are two interpretations over S, then for each individual $e \in S$, $e^{\mathcal{I}_1} = e^{\mathcal{I}_2}$.

already mentioned at the beginning of Section 3.6.1.2, it is assumed that all interpretations over a same signature S share the same infinite domain Δ, and that an individual $e \in S$ is mapped to a same element of Δ by all these interpretations. The first example is the following KB K_1:

Ex 3.6.4.

$K_1 = \{(1)$ $\texttt{class}(\textit{USS Radford DD-120}, \textit{Wickes-class destroyer})$,

(2) $\texttt{class}(\textit{Sphingomonas}, \textit{Alphaproteobacteria})$

(3) $\texttt{class}(\textit{Nymphon}, \textit{Sea spider})$

(4) $\exists \texttt{class}.\top \sqsubseteq \texttt{MeanOfTransportation}$ $\}$

The DL role `class` is understood in two intuitively incompatible ways here, either as specifying the type of military unit (if this type was not reified, the first statement could arguably be replaced by $\texttt{Wickes-class destroyer}(\textit{USS Radford DD-120})$), or in the biological sense (in this case, *Sphingomonas* and *Nymphon* are not understood as instances of *Alphaproteobacteria* and *Sea spider* respectively). Individually though, each statement has a relatively clear meaning. But the KB taken as a whole has at least two obvious non-intended consequences, namely $\psi_1 = \texttt{MeanOfTransportation}(\textit{Sphingomonas})$, and $\psi_2 = \texttt{MeanOfTransportation}(\textit{Nymphon})$. Let us assume that one wants to get rid of these two consequences, i.e. to contract $\mathrm{Cn}(K_1)$ by $\{\psi_1, \psi_2\}$ understood disjunctively, or equivalently to contract $\mathrm{Cn}(K_1)$ by $\psi = \top \sqsubseteq \neg\{\textit{Sphingomonas}\} \sqcup \neg\{\textit{Nymphon}\} \sqcup \texttt{MeanOfTransportation}$.

Let M be the set of all interpretations over $\mathrm{sig}(K_1 \cup \{\psi\}) = \mathrm{sig}(K_1)$, and let us assume that an atom-based representation of the difference between interpretations is adopted. Then it can be shown that for any model $\mathcal{I}_0$ of K_1, each interpretation $\mathcal{I} \in M$ which does not verify ψ and whose difference with $\mathcal{I}_0$ is minimal is such that $\mathrm{diff}(\mathcal{I}_0, \mathcal{I}) = \{\psi_1, \psi_2\}$. Therefore, no matter which one of set

inclusion or cardinality is used, all eligible models closest to $\mathcal{I}_0$ verify the formula $\gamma = \exists \texttt{class}.\top \sqsubseteq \texttt{MeanOfTransportation} \sqcup \neg\{\textit{Sphingomonas}\} \sqcup \neg\{\textit{Nymphon}\}$, which can be paraphrased by "everything which has a class is a mean of transportation, with the exception of *Sphingomonas* and *Nymphon*". Furthermore, these closest models also verify statements 1 to 3 of K_1, such that the output of the contraction process must be equivalent to statements 1 to 3 with γ, no matter whether a local or global ordering of interpretations is used.

This output is clearly not satisfying. Not only do the two contradictory meanings of the role `class` still coexist, but adding to K_1 a third statement about biological *genera* similar to the two previous one (e.g. `class`(*Spirula*, *Coleoidea*)) would yield a new unintended consequence (`MeanOfTransportation`(*Spirula*)). It may also be noted in this case that the recovery postulate for contraction is satisfied by this operation, i.e. extending $\mathrm{Cn}(K_1) - \psi$ with $\{\psi_1, \psi_2\}$ allows for the recovery of $\mathrm{Cn}(K_1)$, such that the problem may not be specific to model-based contraction, but more generally to belief set contraction.

The case of revision is similar. Let $\theta_1 = \neg$`MeanOfTransportation`(*Sphingomonas*), $\theta_2 = \neg$`MeanOfTransportation`(*Nymphon*), and let us assume one wants to revise $\mathrm{Cn}(K_1)$ by $\{\theta_1, \theta_2\}$ understood conjunctively, or equivalently revise $\mathrm{Cn}(K_1)$ by $\theta = \{\textit{Sphingomonas}, \textit{Nymphon}\} \sqcap \texttt{MeanOfTransportation} \sqsubseteq \bot$. The operation yields $\mathrm{Cn}(K_1) * \theta$ equivalent to $(\mathrm{Cn}(K_1) - \psi) + \theta$, with $\mathrm{Cn}(K_1) - \psi$ defined as above.

This leaves one possibility, which is to use of a signature-based representation of the difference between interpretations. The output is slightly more satisfying, yielding a base equivalent to statements 1 to 3. The two meanings of `class` still coexist though, and the consequence `MeanOfTransportation`(*USS Radford (DD-120)*) is lost.

But the following example is an illustration of the inadequacy of model-based con-

traction/revision with a signature-based representation as well. Its syntactic structure is almost identical to the one of example 3.6.4.

Ex 3.6.5.

$K_2 = \{(1)$ $\texttt{award}(\textit{John Goodricke}, \textit{Copley Medal})$,

(2) $\texttt{award}(\textit{Subrahmanyan Chandrasekhar}, \textit{Royal Medal})$,

(3) $\texttt{award}(\textit{Walter Sydney Adams}, \textit{Royal Society})$,

(4) $\top \sqsubseteq \forall \texttt{award}.\texttt{Award}\}$

The problem here is that the Copley Medal and the Royal Medal are both awarded by the Royal Society, but the Royal Society itself is not an award, such that the consequence $\psi = \texttt{Award}(\textit{Royal Society})$ is incorrect. Now consider any model $\mathcal{I}_0$ of K_2, and take the model $\mathcal{I}$ over $\text{sig}(K_2)$ defined by $\texttt{award}^{\mathcal{I}_0} = \texttt{award}^{\mathcal{I}}$, and $\texttt{Award}^{\mathcal{I}} = \emptyset$. If a signature-based representation is adopted, $\text{diff}(\mathcal{I}_0, \mathcal{I}) = \{\texttt{Award}\}$, such that for any $\mathcal{I}'$ over $\text{sig}(K_2)$ with $\mathcal{I} \not\models \psi$, $\text{diff}(\mathcal{I}_0, \mathcal{I}) \subseteq \text{diff}(\mathcal{I}_0, \mathcal{I}')$, and as a consequence $|\,\text{diff}(\mathcal{I}_0, \mathcal{I})| \leq |\,\text{diff}(\mathcal{I}_0, \mathcal{I}')|$. So no matter whether a local or global ordering of interpretations is adopted, one model $\mathcal{I}$ among the selected ones will be such that $\texttt{Award}^{\mathcal{I}} = \emptyset$, and therefore neither $\texttt{Award}(\textit{Copley Medal})$ nor $\texttt{Award}(\textit{Royal Medal})$ is a consequence of the output (axiom 4 will be lost as well). The case of a revision is similar, with $\theta = \neg\texttt{Award}(\textit{Royal Society})$.

If an atom-based representation is adopted instead, the output is only slightly more satisfying, in that the contraction $\text{Cn}(K_2) - \psi$ is equivalent to statements 1 to 3 with $\gamma = \top \sqsubseteq \forall \texttt{award}.\texttt{Award} \sqcup \{\textit{Royal Society}\}$.

The hybrid syntactico-semantic representation adopted for revision by [MLB05] and [QLB06] does not solve the problem either: in both examples, it produces the same output as a revision guided by an atom-based representation of the difference between interpretations.

3.6.2 Syntax-based contraction/revision

This section provides a basic introduction to some of the main notions behind syntax-based revision/contraction in DLs. Algorithmic aspects in particular are omitted on purpose. A more technical and critical state of the art will be provided in Chapter 8, immediately preceding some proposals relevant to this topic.

As opposed to belief set contraction/revision, the field is loosely structured, with contributions apparently pertaining to different disciplines. Some of the most influential proposals were made in the field of *diagnosis* (notably in [Rei87]), whereas other authors, like [RW09] or [QHH+08], explicitly positioned their work wrt the AGM framework, presented in Section 3.6.1. Finally, a large amount of work is issued from the knowledge engineering and DL communities, not explicitly related to belief change, for instance [BP10], and only sometimes to diagnosis, like [Sch05, FS05]. Syntax-based contraction/revision may be designated in these latter communities as “KB debugging” or “syntax-based debugging”. The term "syntax-based contraction/revision" is preferred here, in order to avoid a possible confusion with the techniques described in section 3.4.

A consequence of this diversity is that the terminology may largely vary from one author to the other. But because the notions are essentially identical, the choice is made here to propose a unified view, relying for readability on a unique terminology, which can partly be viewed as a simplification of the one adopted in [RW09].

Syntax-based contraction/revision will be put into perspective with two other sections of this chapter: first with the belief set contraction/revision scheme introduced in Section 3.6.1, and then with justification-based debugging presented in Section 3.4.

3.6.2.1 Belief bases VS belief sets

Syntax-based contraction/revision or *belief base* contraction/revision consists in discarding some axioms from an (not necessarily deductively closed) input KB K in order to get rid of some undesired consequence ψ in the case of contraction, or to restore the consistency of $K \cup \{\theta\}$ while preserving θ (if $\{\theta\}$ is consistent) in the case of revision. Another way to view it is that base contraction (resp. revision) is a contraction of $\mathrm{Cn}(K)$ by ψ (resp. a revision of $\mathrm{Cn}(K)$ by θ) with the additional requirement that the output KB must be a syntactic subset of K (resp. a syntactic subset of K extended with $\{\theta\}$), and not only a weaker version of it.

As opposed to operations on belief sets, the distinction between contraction and revision is not as relevant here, because a revision of a (finite) base K by a formula θ can be reduced to the contraction of $K \cup \{\theta\}$ by any contradiction (e.g. by $\top \sqsubseteq \bot$ in DLs), with the additional constraint that θ should be preserved. This was not the case for belief sets, because the input base of a contraction operation was viewed as a theory.

The generalization to sets of formulas is straightforward, which compensates for the fact that most DLs are not closed under conjunction or disjunction. For the contraction of an input KB K by a set of formulas Ψ, the only decision to make is whether Ψ is understood conjunctively or disjunctively. In the former case, a candidate output R is a syntactic subset of K such that $\Psi \not\subseteq \mathrm{Cn}(R)$. In the latter case, a candidate output R is a syntactic subset of K such that $\Psi \cap \mathrm{Cn}(R) = \emptyset$. For a revision of K by Θ, Θ is generally understood conjunctively (or at least it will be here). So if K and Θ are both consistent (which is the only non-trivial case), a candidate output R is a syntactic subset of K such that $R \cup \Theta$ is consistent. The following notation is adopted throughout this thesis. When contracting K by Ψ

(resp. when revising K by Θ), $\mathcal{R}$ will designate all admissible subsets of K, i.e. $\mathcal{R} = \{R \subseteq K \mid R \not\vdash \bot\}$ (resp. $\mathcal{R} = \{R \subseteq K \mid R \cup \Theta \not\vdash \bot\}$).[11]

This very broad definition does not provide any guarantee of minimal information loss. A common additional requirement is that for each candidate output base, the number of discarded axioms should be minimal wrt set inclusion. In other words, the typical set of candidate output subbases is $\mathcal{R}_{\subseteq} = \max_{\subseteq} \mathcal{R}$. This can be seen as a coarse-grained form of minimal information loss (as opposed to belief set revision), but with some non-negligible practical advantages.

In example 3.6.4 Section 3.6.1.2.2, the two candidate output subbases for contraction are {1,4} and {1,2,3}. In example 3.6.5, these are {1,2,3} and {1,2,4}. The case of 3.6.4 is debatable, but for 3.6.5, there is one satisfying solution among the candidate outputs, namely {1,2,4}. In general, belief base contraction/revision results in a more important information loss than belief set contraction/revision, but as illustrated by these examples, there is also often a satisfying solution among the candidate output bases. Example 3.6.5 also illustrates one of the main practical issues of belief base contraction/revision, which is the multiplicity of candidate output bases, some of which (namely {1,2,3} here) may be clearly inappropriate. This is one of the problems addressed by the proposals which will be made in Chapter 8.

Another argument for base contraction/revision in a Semantic Web context is traceability. The belief set setting completely abstracts from the syntax, and therefore offers no guarantee about the syntactic formulation of the output KB. On the other hand, base contraction guarantees that all axioms of the output KB are also axioms of the input KB. More generally, it is important to understand that outside of propositional logic, most belief change operators on belief sets are only characterized in terms of intuitive and philosophically grounded constraints (the postulates), but

[11] Again, $X \vdash \bot$ is a shortcut for $X \vdash \top \sqsubseteq \bot$, as explained in Chapter 2 Section 2.3.6.

rarely implemented.

[RW09] adapted the terminology used for belief set contraction/revision to belief base contraction/revision, and fully characterized these operations in terms of postulates for bases, inspired by the ones for beliefs sets. In particular, they define six different types of contraction/revision operators satisfying different sets of postulates. For readability, this formal characterization is not fully adopted here, neither the exact notation, but only the notions needed for the work presented in this thesis. In particular, just as for belief sets, most postulates deal with limit cases, and are needed for proofs only. Another reason why a simplified notation is sufficient is that as opposed to the literature on belief sets, which is arguably richer in paradoxes and potentially counterintuitive results, the literature on belief bases relies on relatively straightforward intuitions.

By analogy to belief set contraction/revision, the family $\mathcal{R}_{\subseteq}$ is called the *remainder set* in [RW09], and here as well, and each element of $\mathcal{R}_{\subseteq}$ is called a *remainder*.[12] In practice, the size of the (base) remainder set $\mathcal{R}_{\subseteq}$ may be important. More exactly, it is bounded by the size of the largest subset $\mathcal{W}$ of 2^K such that for all $W_1, W_2 \in \mathcal{W}$, if $W_1 \neq W_2$, then $W_1 \not\subseteq W_2$, i.e. $\binom{|K|}{\lceil \frac{|K|}{2} \rceil}$. Additionally, as illustrated by example 3.6.5, some elements of $\mathcal{R}_{\subseteq}$ may be more relevant than others. As for belief sets, a *selection function* $\sigma : 2^{2^{\mathcal{L}}} \mapsto 2^{2^{\mathcal{L}}}$ may select some of these remainders only. Then by default, and in virtue of the abovementioned determinism principle, the output of the debugging process may be the intersection of these selected remainders, i.e. $\bigcap \sigma(\mathcal{R}_{\subseteq})$ (note that $\bigcap \sigma(\mathcal{R}_{\subseteq}) \in \mathcal{R}$). As opposed to belief set contraction though, σ may select only one remainder, and ideally selects (deterministically) a small number of them,

[12] Formally, a remainder set is traditionally defined (for contraction) as a function of K and Ψ, but this simplified notation will be without ambiguity here, and is also convenient in order to compare different approaches to KB debugging, as will appear in Chapter 8

in order to lose as few axioms as possible. This operation was fully characterized in [RW09] in terms of postulates for bases.

But alternatively, the output of the debugging process may be a so-called *disjunctive knowledge base*, introduced for DLs by [MLB05], which can be intuitively understood as the disjunction of all selected remainders. If $\mathcal{X}$ is a family of sets of formulas, then $\bigvee \mathcal{X}$ will designate the disjunctive KB built out of $\mathcal{X}$. Because most DLs are not closed under disjunction, this construction can generally not be natively represented. But it can be simulated with a set of KBs, requiring that $\mathrm{Cn}(\bigvee \mathcal{X}) = \bigcap_{X \in \mathcal{X}} \mathrm{Cn}(X)$, or in other words that a formula ψ is a consequence of $\mathrm{Cn}(\bigvee \mathcal{X})$ iff it is a consequence of each KB in $\mathcal{X}$. Then $\bigvee \sigma(\mathcal{R}_{\subseteq})$ is arguably a better output than $\bigcap \sigma(\mathcal{R}_{\subseteq})$, because it results in a less important information loss, without compromising determinism. In practice though, this construction is not well adapted to the Semantic Web framework, which relies on the possibility of importing axioms from single KBs. More generally, even in a closed environment, maintaining a family of KBs instead of a single one may be complex from an engineering point of view.

For the concrete belief base contraction/revision algorithms proposed in this thesis, if a family $\sigma(\mathcal{R}_{\subseteq})$ of several remainders is selected, the choice was made to remain agnostic wrt to the form of the final output of the process. Depending on the applicative requirements, the user may decide to keep the intersection $\bigcap \sigma(\mathcal{R}_{\subseteq})$ of these subbases, their disjunction $\bigvee(\sigma\mathcal{R}_{\subseteq})$, or even to review them manually (or more realistically their respective complements in K), and select the most appropriate ones.

3.6.2.2 Diagnoses and justifications

Section 3.5 introduced the most common techniques to compute *justifications* for a consequence ψ of an input KB K. Justifications are minimal subsets of K (wrt set inclusion) entailing ψ. Computing all justifications for ψ in K is actually closely related to the problem of computing the (base) remainder set $\mathcal{R}_{\subseteq}$ when contracting K by ψ, i.e. the family of all maximal subbases of K not entailing ψ, as will be explained in this section.

A *diagnosis* or *minimal incision* for K and ψ is a minimal subset D of K such that $(K \setminus D) \not\models \psi$. Once again, minimality is understood wrt set inclusion, i.e. the set of all diagnoses for K and ψ is $\mathcal{D} = \min_{\subseteq}\{\Delta \subseteq K \mid (K \setminus \Delta) \not\models \psi\}$. Equivalently, the complement R of a diagnosis D in K is a maximal subset of K (wrt set inclusion) such that $R \not\models \psi$, or in other words, if $\mathcal{R}_{\subseteq}$ is the (base) remainder set for K and ψ, then D is a diagnosis iff $D \subseteq K$ and $K \setminus D \in \mathcal{R}_{\subseteq}$.

Computing diagnoses can be viewed as a more automated debugging strategy than computing justifications. A justification is a possible explanation for ψ to hold, whereas a diagnosis is a suggestion of axioms to be discarded to get rid of ψ.

Justifications and diagnoses are also closely related notions. If $\mathcal{J}$ is the set of all justifications for ψ in K, then the family $\mathcal{D}$ of all diagnoses for ψ in K is the set of all *minimal hitting sets* (wrt set inclusion) for $\mathcal{J}$. In other words, $D \in \mathcal{D}$ iff D has an nonempty intersection with all elements of $\mathcal{J}$ and is minimal wrt set inclusion, i.e. $\mathcal{D} = \min_{\subseteq}\{\Delta \subseteq K \mid \text{for all } J \in \mathcal{J},\ J \cap \Delta \neq \emptyset\}$. Intuitively, discarding one axiom from each $J \in \mathcal{J}$ is sufficient and necessary to get rid of ψ, and discarding any additional axiom would result in some unnecessary syntactic information loss. Note that if $\mathcal{J} = \{J_1, .., J_n\}$, because justifications may overlap, it may be the case that $|\mathcal{D}| < |J_1 \times .. \times J_n|$. Conversely, even if this may be less intuitive at first sight,

each justification $J \in \mathcal{J}$ is a minimal hitting set for $\mathcal{D}$, i.e. $\mathcal{J} = \min_{\subseteq}\{\Delta \subseteq K \mid \text{for all } D \in \mathcal{D},\ D \cap \Delta \neq \emptyset\}$. So computing $\mathcal{D}$ out of $\mathcal{J}$ (resp. $\mathcal{J}$ out of $\mathcal{D}$) amounts to computing all minimal hitting sets for $\mathcal{J}$ (resp. $\mathcal{D}$).

Another way of viewing this correspondence between $\mathcal{J}$ and $\mathcal{D}$ leads back to the pinpointing formula α, introduced in Section 3.5.1.2, used to compute all justifications with a glass-box algorithm. Each prime implicant of α is a justification, and each prime implicate of α is a diagnosis. So in the worst case, computing all justification or all diagnoses from a pinpointing formula are both as hard. But in practice, as will be explained in Chapter 8 Section 8.3.2, if a saturated tableau algorithm with a tracing mechanism is used to compute $\mathcal{J}$ or $\mathcal{D}$, depending on whether $\mathcal{J}$ or $\mathcal{D}$ is the desired output, the computational cost of each operations for a same input KB may differ.

An alternative strategy to compute diagnoses is Reiter's algorithm, proposed in [Rei87], and corrected in [GSW89]. The correspondence between diagnosis and base contraction/revision was established by [Was00], and different variations of Reiter's algorithm have been implemented to compute diagnoses in DLs, for instance by [Sch05], [FS05] or [Kal06]. A detailed presentation of Reiter's algorithm will be provided in Chapter 8 Section 8.3.1. It proceeds by expanding a directed acyclic graph in a breadth-first fashion, with a single root node. Each node is labeled with a set of axioms, and each edge with one axiom. After termination, each path from the root node to a leaf node is a diagnosis, and conversely. In addition, each node label of the graph is a justification (or $\emptyset$, by convention), and conversely.

As will be proven in Chapter 8 Section 8.3.1, and as opposed to glass-box algorithms, computing $\mathcal{J}$ or $\mathcal{D}$ with Reiter's algorithm are closely related tasks, in that it cannot be guaranteed that any of the two has actually been computed if the other one has not been computed as well during the execution. Therefore the cost of both

operations is identical, not only in the worst-case, but for any input KB.

It turns out that most practical attempts to compute diagnoses for DLs with Reiter's algorithms only produce some of them though. Both [Sch05] and [FS05] for instance only compute diagnoses up to a certain size. Several other strategies to avoid the computation of the whole $\mathcal{D}$ have been proposed or implemented, for instance by [Kal06] or [QHH+08]. Some of these approaches do not compute diagnoses in practice, but supersets of diagnosis. A more technical review of these techniques is provided in Chapter 8.

The notion of diagnosis can be generalized in a straightforward way to a set of consequences Ψ, as previously for justifications and remainders, where Ψ may be understood either conjunctively of disjunctively.

Finally, the notion of diagnosis also applies to the case of revision, which is actually more faithful to its initial formulation by [Rei87]. The relation between diagnosis and justifications in this case is slightly different. If a consistent base K is to be revised by a consistent set Θ of axioms, and if $\mathcal{J}$ is the family of all justifications for the inconsistency of $K \cup \Theta$, then D is a diagnosis iff it is a minimal hitting set for the family $\mathcal{V} = \min_{\subseteq}\{J \cap K \mid J \in \mathcal{J}\}$.

3.7 Conclusion

This Chapter introduced some of the main paradigms and algorithms of the KB debugging literature, with a strong an emphasis on DLs.

Approaches based on the review of admissible models of the input KB aim at identifying missing information, and eventually at strengthening the KB, which fundamentally differs from the objective being pursued here, namely weaken a KB in order to get rid of nonsense.

Approaches based on formal ontology are characterized by their relatively low automation. In particular, the required manual ontological analysis of the input KB may be costly. As an alternative, Chapter 7 will discuss the efficiency of a manual but fast and shallow ontological extension instead, possibly coupled with an automated debugging strategy, presented in Chapter 8.

Pattern-based approaches to KB debugging on the other hand rely on some extended forms of syntactic typos, which unfortunately are often too generic to spot the type of errors targeted here, i.e. the coexistence of incompatible meanings of an individual or predicate within a KB. This legitimates the introduction of external sources of knowledge to the debugging process, for instance linguistic evidence, as investigated in chapters 4 and 6.

Belief change offers at first sight a more promising formal apparatus for the objective being pursued here. But formula-based contraction/revision, introduced in Section 3.6.1.1, mostly provides formal results, and not actual algorithms. In addition, it is shown is Section 3.6.1.1.2 that in the case of revision, for some DLs at least, this paradigm does not currently provide an effective principle of minimal change. Model-based contraction/revision is introduced in Section 3.6.1.2, which also shows why proposals made in this field may not be adapted to the problem at hand, due in particular to their close relatedness to default reasoning.

The last paradigm investigated in this chapter is syntax-based contraction/revision in Section 3.6.2, which aims at identifying sets of axioms to be preferably discarded from an input KB, together with the closely related problem of computing justifications in Section 3.5. These two sections introduce some of the main notions used in the field (justification, diagnosis, ...), which will be used intensively in Chapters 6 and 8. The main (black-box and glass-box) algorithms are also briefly presented (and will be discussed in more details in Chapter 8). Syntax-based contraction/revision

generally suffers from the number of candidate output subbases, which is why Chapter 8 studies the automated computation of a selection of them only, possibly guided by linguistic evidence.

Chapter 4

Linguistic evidence for KB debugging

This chapter investigates the integration of automatically gathered linguistic evidence to a KB debugging process. Natural Language Processing (NLP) for knowledge engineering has been extensively studied in the fields of information extraction and ontology learning from texts. But to our knowledge, relying on linguistic evidence to detect and/or repair errors in an input KB is an original proposal, as most works in these fields focus either on extending an existing KB, or learning one from scratch.

This may also be viewed as a small paradigm shift: data extracted from texts is generally viewed as less reliable than manually crafted knowledge. But as illustrated in Chapter 1, when a KB grows in size, it also tends to contain sets of axioms which may make sense individually, but violate elementary common sense when considered together. This phenomenon is exacerbated in a Semantic Web context, where the KB may integrate data from multiple sources (some of which may actually have been automatically extracted from unstructured or semi-structured texts). The hypothesis

is made here that linguistic evidence may be a relevant common ground in order to decide among incompatible conceptualizations. More exactly, the output KB of a debugging process, viewed as a logical theory, should reflect some statistically meaningful regularities observed in a linguistic corpus.[1]

Section 4.1 offers a broad typology of NLP techniques used in information extraction or ontology learning, and explains why some of them are not practically applicable to debugging. This legitimates a focus on named entities, as well as the usage of distributional evidence, both choices being discussed in Section 4.2. This section also contains the main contribution of this chapter, which is an original framework in order to evaluate to what extent a candidate output base Q of the debugging process is in line with distributional evidence, or equivalently, to what extent some linguistic input may indicate that Q contains violations of common sense of the type presented in Chapter 1. Finally, Section 4.3 is a relatively high-level introduction to distributional similarity, paired with the description of the specific distributional settings adopted in this work.

4.1 NLP for information extraction and ontology learning

The exploitation of NLP techniques in knowledge engineering has been mostly investigated in the fields of information extraction and ontology learning from texts. This

[1] An arguably better high-level formulation of this idea is that the knowledge expressed by the KB should be as "consistent" as possible with the linguistic input, where "consistent" is understood in its vernacular sense, and not its logical one. This formulation is avoided here in order to prevent possible confusions. The term "coherence" is not used either, for the same reason (see Chapter 2 Section 2.3.5). In particular, for lack of a better term, "compliance" will sometimes be used in order to designate the degree to which a given KB is in line with the linguistic input.

section offers a selective overview of tasks and techniques pertaining to these fields, and explains in particular why several of these techniques may not be appropriate for KB debugging, or hardly applicable.

4.1.1 Cooccurrence patterns for relation extraction

Relation extraction generally deals with the identification of semantic binary relations holding between two linguistic terms. Each of these these linguistic terms may denote either an individual (for instance the term "Woody Allen") or an atomic concept (e.g. "festival").

One of the most widespread approach to binary relation extraction relies on cooccurrence patterns, which may be manually crafted or automatically learned from known instances. For instance, the pattern "$< X_1 >$s and other $< X_2 >$s" is likely to indicate that the concept denoted by X_1 is subsumed by the concept denoted by X_2, i.e if A_i is the atomic concept denoted by the term X_i, then $A_1 \sqsubseteq A_2$ is likely to hold.

The most influential illustration of this strategy is probably the work of [Hea92], who sketches a bootstraping approach to the extraction of subsumptions based on such cooccurrence patterns. The algorithm starts with a small set of manually crafted cooccurrence patterns like the one above, and these patterns are projected on large corpora in order to retrieve pairs of terms (denoting concepts which are) likely to instantiate the relation. Then new patterns can be identified out of other cooccurrences of these pairs of terms, and in turn projected to learn new pairs instantiating the relation, etc. Alternatively, the loop may be initiated with a set of pairs of terms, instead of a set of patterns.

The two most studied relations are by far subsumption ($A_1 \sqsubseteq A_2$ in DLs , where

A_1 and A_2 are atomic concepts) and instantiation ($A(e)$ in DLs, where A is an atomic concept and e an individual). But these techniques have also been applied to extract instances of other binary relations, like meronymy (part-of), or even arbitrary relations between concepts/individuals. Influential works for these other relations are the ones of [BC99] or [PP06]. Other works rely only partly or indirectly on cooccurrence patterns, such as [SB05] for the extraction of domain specific relations, or [VVSH07] for disjointness axioms (i.e. axioms of the form $A \sqsubseteq \neg B$, with A and B atomic concepts).

Integrating the extracted pairs to a KB generally requires an additional and more or less automated phase of filtering or reformulation, because of the noise (like [NVF11] for the subsumption relation), but also because the semantic interpretation of the extracted relations may be ambiguous. The distinction between concepts and individuals in particular is not always clear. Another case of ambiguity is the interpretation of some relations (other than subsumption or disjointness) holding between concepts. Relation extraction systems prototypically produce triples, for instance $< A_1$, part-of, $A_2 >$, with A_1 and A_2 atomic concepts. But as explained in [VA07] for this specific case (meronymy), quantification remains ambiguous, i.e. this triple may correspond to different DL axioms. Either the whole implies the existence of the part (like in `Handle/Door`, i.e. `Door` $\sqsubseteq \exists$`hasPart.Handle` in DLs), or the converse (like in `Airbag/Vehicle`, i.e. `Airbag` $\sqsubseteq \exists$`partOf.Vehicle` in DLs), or both (like in `Letter/Alphabet`, i.e. `Alphabet` $\sqsubseteq \exists$`hasPart.Letter` and `Letter` $\sqsubseteq$ $\exists$`partOf.Alphabet` in DLs).

But the main reason why cooccurrence patterns may not be appropriate for KB debugging is a different one. In a debugging scenario, the signature $\text{sig}(K)$ of the input KB K is known in advance. So let us assume that one wants to confirm or infirm the hypothesis that a given relation holds between two elements e_1 and

e_2 of sig(K), and that a set of linguistic patterns corresponding to this relation is available (possibly learned from other pairs instantiating that same relation according to K). Then a sufficient number of linguistic cooccurrences of terms denoting e_1 and e_2 is required to verify whether or not they appear together with these patterns. Alternatively, one may use a set of patterns indicating that this relation does not hold, but the problem remains identical: even with the whole web available, it is very unlikely to find a sufficient number of cooccurrences of these two terms. The problem does not hold for ontology learning/population, because the projection of a cooccurrence pattern can return any pair of terms, regardless of the signature of the KB under construction. For instance, the cooccurrence "trumpets and other brass instruments" may be relevant for ontology learning, even if neither `Trumpet` nor `Brass Instrument` is a concept of the KB under construction yet. But for debugging, if any of the two is not in sig(K), this cooccurrence is meaningless.

This is why the approach presented in Section 4.2 relies on *simple occurrences* of target terms, and not on *cooccurrences* of pairs of target terms. For instance, in order to determine whether the individual *Thaddeus S.C. Lowe* is likely to be an instance of the atomic concept `Person`, the approach will use simple occurrences of the term "Thaddeus S.C. Lowe", and intuitively look for contexts likely to indicate that Thaddeus S.C. Lowe is indeed a person (like "Thaddeus S.C. Lowe was born") or is not (like "located in Thaddeus S.C. Lowe"). On the other hand, a relation extraction approach based on cooccurrence patterns would require cooccurrences of the terms "Thaddeus S.C. Lowe" and "person", which are very unlikely to be found in sufficient quantity. Querying the web with instantiated cooccurrence patterns would not solve the problem either, which can be easily verified empirically. For instance, none of the pages indexed by Google contains any of the following strings: "Thaddeus S.C. Lowe and other persons", "Thaddeus S.C. Lowe, the first/last/only

person", "Thaddeus S.C. Lowe is/was a person", "Thaddeus S.C. Lowe, a person", etc.[2]

4.1.2 Fine-grained axiom extraction from definitorial or encyclopedic context

A different category of works aim at extracting OWL TBox axioms from natural language texts almost literally, based on a tight correspondence established between some OWL constructs and natural language syntax. A classical example is the noun modifier syntactic function: for instance, if "epilepsy" denotes a concept, then "myoclonus epilepsy" is likely to denote a subclass of it, and "progressive myoclonus epilepsy" a subclass of the latter, etc.

But more sophisticated OWL TBox axioms may also be extracted, prototypically out of syntactically parsed definitions. A good illustration is the work presented in [VHC07]. For instance, out of the definition "Data: facts that result from measurements or observations", the axiom extraction approach developed by the authors successfully produces $\texttt{Data} \equiv \texttt{Fact} \sqcap \texttt{resultsFrom}.(\texttt{Measurement} \sqcup \texttt{Observation})$, thanks to a tight mapping from syntactic functions (object, modifier, ...) and function words ("that","or",...) to OWL constructs.

A first limitation is the reliability of such a transformation from a syntactically analyzed definition to an axiom. Among others, it suffers from prepositional attachment ambiguities, as well as the interpretation of these prepositions, and more generally from the polysemy of function words. Here are two other examples given by the same authors:

"Currency: a currency is a unit of exchange, facilitating the transfer of goods and

[2] If the content of this thesis gets indexed, it may actually provide the first occurrences.

services."⇒

$\texttt{Currency} \equiv \texttt{Unit} \sqcap \exists \texttt{of}.\texttt{Exchange} \sqcap \exists \texttt{facilitate}.(\texttt{Transfer} \sqcap \exists \texttt{of}.(\texttt{Good} \sqcap \texttt{Service}))$

"Biosphere: the portion of Earth and its atmosphere that can support life."⇒

$\texttt{Biosphere} \equiv \texttt{Portion} \sqcap \exists \texttt{of}.((\texttt{Earth} \sqcup (\texttt{Its} \sqcap \texttt{Atmosphere})) \sqcup \exists \texttt{canSupport}.\texttt{Life})$

Among the difficulties illustrated by these examples are the interpretation of "and" (nothing is meant to be both a good and a service in the first definition), the debatable interpretation of "of" in both definitions (independent from "unit" in the first axiom, from "portion" in the second), and the underspecification of the possessive "its" (the possessor may be either the biosphere or the earth). So although the approach seems useful as a pre-extraction step, an additional manual correction phase is required if these axioms are to be integrated into a KB.

But more fundamentally, just as in the previous section, these fine-grained extraction approaches seem hardly applicable to KB debugging because the signature of the input KB is known in advance, whereas in an ontology learning context, it can be freely extended with concepts and roles like `Exchange`, `canSupport` or even `of`.

Finally, another important practical limitation comes from the fact that the corpus must be of an encyclopedic nature, because terms denoting predicates or named entities are only marginally defined outside of manuals, dictionaries or encyclopedias. For instance, it is relatively uncommon to read a text explicitly stating that "a car is a vehicle", or that "seats are parts of a car". These informations are more likely to be conveyed indirectly through non definitorial uses of the term "car", for instance by the expression "drive his/her car" or "sitting in his/her car", etc. This is one of the motivations behind approaches based on linguistic term distribution.

4.1.3 Taxonomy induction based on term distribution

Taxonomy induction from texts is generally defined as the extraction of a set of axioms of the form $A \sqsubseteq B$, where A and B are atomic concepts. Aside from the exploitation of cooccurrence patterns, presented in Section 4.1.1, part of the literature on the topic focuses on the distributions of terms denoting these concepts, based on the assumption that terms which share similar linguistic contexts tend to have similar meanings.

[Cim06] implemented a clustering approach inspired by formal concept analysis (FCA) for this purpose. FCA is a taxonomy induction technique which can intuitively be viewed as a reverse application of the principle of property inheritance used (among others) in object-oriented programming. Given two finite sets O and P of objects and properties respectively, an FCA concept is a pair $< O', P' >$ such that $O' \subseteq O$, $P' \subseteq P$, O' is exactly the set of elements of O verifying all properties in P', and P' is exactly the set of elements of P verified by all elements of O'. Then the subsumption relation between concepts is given by $< O'_1, P'_1 > \sqsubseteq < O'_2, P'_2 >$ iff $O'_1 \subseteq O'_2$ iff $P'_2 \subseteq P'_1$. For instance, if $< O'_2, P'_2 >$ is the `Animal` concept, then $< O'_1, P'_1 >$ may be `Bird`. All birds are animals ($O'_1 \subseteq O'_2$) but all animals are not birds ($O'_2 \not\subseteq O'_1$), and birds verify all properties verified by animals ($P'_2 \subseteq P'_1$), but also properties that some animals do not verify, e.g. they can fly ($P'_1 \not\subseteq P'_2$). [Cim06] adapted this framework to taxonomy induction from texts, using common nouns as FCA objects, and linguistic contexts as FCA properties. He gives table 4.1.3 as an example. All six linguistic terms in the left column may appear as direct objects of the verb "book", which is why they all have the property of being bookable. But only two of them ("excursion" and "trip") are likely to appear as direct object of the verb "join". This yields a small taxonomy, where for instance the concept of bookable,

	bookable	rentable	driveable	ridable	joinable
hotel	✓				
apartment	✓	✓			
car	✓	✓	✓		
bike	✓	✓	✓	✓	
excursion	✓				✓
trip	✓				✓

Table 4.1: Concept hierarchy induction with FCA, reproduced from [Cim06]

rentable and driveable things (cars and bikes) is subsumed by those which are only bookable and rentable (cars, bikes and apartments), whereas the concept of bookable and joinable things (trips and excursions) does not subsume neither is subsumed by any of the two. At the bottom of the taxonomy lie the concepts denoted by each term ("hotel", "apartment", "bike", etc).

A first similarity between this taxonomy induction method and the approach adopted in Section 4.2 is that it relies on a linguistic phenomenon known as *selectional preference*. The linguistic context ("book" + obj, "drive"+obj) retained for the taxonomy induction are not any contexts appearing with each of these six terms, but only contexts whose occurrences with these terms is statistically meaningful. For instance, the observation that two target terms may appear as objects of the verb "have" is intuitively less meaningful than objects of the verbs "book" or "drive", because of the overall frequency of "have". But the distributions of the objects of a verb also comes into play. For instance, it may be expected that the distribution of objects of a verb like "drive" is more informative than the distribution of objects of "describe" or "present", because the two latter may take almost any common noun as an object, not the former. Or in information-theoretic terms, the entropy of a random variable modeling the distribution of the objects of "drive" (i.e the expected number of occurrences of "car", "bottle", etc. as the object of "drive" for

n occurrences of "drive" with an object in natural language texts) should probably be lower than for "describe" or "present". The corresponding linguistic phenomenon is known as *selectional preferences*, and was notoriously studied in NLP by [Res97]. Informally, selectional preferences model the fact that a term t_1 (a verb, but also a preposition, an adjective, a verb+preposition, ...) may (or may not) constrain the semantic type of a term t_2 in a given syntactic position (e.g. object, modifier, ...). For instance, "located in $< X >$" tends to select the semantic category `Place` for X. As suggested by the "drive" example, the selected semantic categories are not limited to a few very abstract types (like `Physical entity`, `Event`, etc), but may actually be more specific. In distributional semantics, the intuition behind selectional preferences is traditionally captured by frequency weighting techniques, introduced in Section 4.3.4.

Another interesting particularity of this taxonomy induction by [Cim06] is that aside from the most specific ones, the concepts of the inferred taxonomy do not have a linguistic label, but simply correspond to a set of linguistic contexts (the FCA properties). For instance, the concept of bookable and joinable thing is not labeled by a linguistic term. This is another similarity with the approach adopted in Section 4.2, which in a sense goes one step further into that direction, by completely abstracting from concept labels, using individuals labels instead as target terms.

An important limitation of this approach though is the requirements put on the corpus. For the FCA approach to be effective without manual completion, the exhaustive set of admissible linguistic contexts for each target term must be know, i.e. the corpus must contain a (statistically meaningful) trace of this possibility, otherwise the whole taxonomy will be affected. For instance, in the above example, although bikes are certainly bookable things, let us assume that no occurrence of the verb "book" with "bike" as a direct object was found in practice in the corpus. This

will immediately affect the structure of the taxonomy, with a new concept of things (namely bikes) which are only rentable, driveable and rideable. In other words, in the absence of explicit evidence for a context c to be applicable to a term t, it is assumed by default that it is not (this can be viewed as a form of closed-world reasoning). As a consequence, for the approach to be effective, a statistically meaningful number of occurrences of t with c must appear in the corpus, which is a very optimistic assumption.

4.1.4 Ontology population

Ontology population deals with the automated instantiation of predicates from an input TBox. As such, it overlaps with some of the relation extraction techniques presented in Section 4.1.1. The focus will be put here on the instantiation of unary FOL predicates, i.e. atomic DL concepts. The task is prototypically a supervised one.[3] Given an annotated corpus where some named entities are tagged with concepts (for instance "Woody Allen" may be tagged with `Director`, or `Person`, or both), the objective is either to retrieve other instances of these concepts in non-annotated corpora, or to classify a given set of individuals according to these concepts, based on linguistic evidence.

The former problem is by far the more often addressed in the literature. It is also closely related to an information extraction task known as named entity recognition and classification (NERC), popularized in the 90's with the MUC evaluation campaigns, and later integrated to several Semeval campaigns. The terminological distinction between NERC and ontology population is not always clear, but

[3]The meaning of "supervised" here is the one traditionally used in machine learning. A supervised task consists in learning a function from a training set of labeled data, as opposed to unsupervised learning, whose goal is to identify a structure (for instance clusters) in some unlabeled dataset.

they tend to differ on the concepts being used, traditionally a small number of relatively abstract semantic types for NERC (prototypically `Event`, `Place`, `Date`, `Person`, `Organization`, etc), whereas ontology population may deal with arbitrarily fine-grained concepts (like `Cruise Ship`, `Volcano`, `NGO`, `Director`, etc). They also differ on the fact that NERC aims at classifying each occurrence of a term independently.

The second problem (classifying a set of already identified individuals, based on linguistic evidence) is more immediately useful for the problem at hand here. The difference with the previous one is that no entity recognition phase is required, or in other words, that the whole signature of the KB is known in advance, which is the case in a KB debugging setting. This is why the emphasis is put here on recent works addressing this second problem, even though it remains a marginal one in the ontology population literature.

[TM08] compare three different approaches to solve a problem coined as follows: given a (finite) set *Con* of mutually disjoint atomic DL concepts with labels, and a (finite) set *Ind* of individuals with labels, assign to each $e \in Ind$ the correct $C \in Con$. The assumption is also made that *Con* covers all elements of *Ind*, i.e. that each $e \in Ind$ is an instance of some $C \in Con$. Finally, a training set of reliable instances of each $C \in Con$ may be used, and these instances do not appear in *Ind*.

The first of the three approaches compared by [TM08] is inspired by [Hea92], and is a simplified version the relation extraction strategy based on cooccurrence already described in Section 4.1.1. It relies on a set of cooccurrence patterns (like "$< X_1 >$ is a $< X_2 >$" or "such $< X_2 >$ as $< X_1 >$"), but they are all manually crafted, and not dynamically learned like in [PP06], such that the training set in this case was not used. Unsurprisingly, the approach suffers from low recall.

The second approach is also unsupervised, but relies on distributional similarity

(for an introduction to distributional similarity, see Section 4.3 below). It was initially proposed by [CV05]. A vector $v(C)$ is build for each $C \in Con$, representing the contexts in which the label of C is observed in the corpus (for instance, a vector for the concept `Director` out of the occurrences of the word "director" in the corpus), and similarly a vector $v(e)$ for each individual $e \in Ind$ (for instance, a vector for the individual *Woody Allen* out of the occurrences of the term "Woody Allen" in the corpus). Then an individual e is assigned to a class C iff the similarity between $v(C)$ and $v(e)$ is maximal among all $C \in Con$. This approach suffered from a relatively low precision. As noted by [TM08], the assumption that the label of a class and the labels of its instances should have similar linguistic behavior is also relatively implausible linguistically. Another limitation of this strategy, discussed in Section 4.2.1.1, is that concept labels in a KB rarely correspond to linguistic terms denoting these concepts in natural language texts.

The third approach is supervised, and also relies on distributional semantics. As opposed to the previous ones, labels of concepts are not used in this approach, but only labels of individuals, which are prototypically proper names, and therefore arguably less ambiguous (the problem of labels' polysemy will be discussed in more details in Section 4.2.1.1). A distributional vector $v(C)$ is built for each $C \in Con$ out of the training set by aggregating the vectors of the known instances of C. Then as previously, a vector $v(e)$ is built for each $e \in Ind$ as well, and e is assigned to a class C iff the similarity between $v(C)$ and $v(e)$ is maximal for $v(e)$ among all $C \in Con$. [TM08] showed that this approach largely outperformed the other two for the benchmark used in their experiments.

[GG08] formulated the ontology population problem in almost identical terms. They use a taxonomy of concepts instead of a set of concepts, but the concepts of Con are only the most specific ones in this taxonomy, and are also supposed

mutually disjoint, such that *Con* for [GG08] is actually similar to *Con* for [TM08]. Their approach relies on the substitution of named entities within linguistic contexts, and also requires a training set of known instances of each $C \in Con$. Let $e \in Ind$ be a named entity to be classified. A set M_e of linguistic contexts in which the label of e appears is extracted from snippets, using a web search engine. These contexts are n-grams (i.e. sequences of n words) which precede, follow or surround an occurrence of the label of e, together with the position of the label. For instance, if e is *Woody Allen*, i.e. if *Woody Allen* is an individual to classify (or in other words *Woody Allen* $\in$ *Ind*), and if its label is "Woody Allen", the context "$< X >$ is currently filming" may be extracted from web snippets retrieved by a search engine for the query "Woody Allen". Then in order to determine whether e is likely to be an instance of a concept C, for each $m \in M_e$, and for each known instance e' of C according to the training set, the label of e' is substituted for $< X >$ in m, and the Web IT 5-gram corpus is searched for occurrences of the resulting n-gram.[4] For instance, if *Michael Haneke* is known to be an instance of the concept `Director`, and if the sequence "Michael Haneke is currently filming" can be found in the the Web IT 5-gram corpus, this would support the hypothesis that *Woody Allen* is also an instance of `Director`. Based on these observations, a score for each hypothesis (`Director`(*Woody Allen*), `Rugbyman`(*Woody Allen*), etc.) is computed, which also takes into account the frequencies of all e' and m within the the Web IT 5-gram corpus, such that observing e' and m is more meaningful if e' and m are less frequent. Then only the hypothesis with the highest score for e (e.g. `Director`(*Woody Allen*)) is retained.

[4] the Web IT 5-gram corpus is an index of n-grams observed in a very large collection of web pages together with their frequencies, released by Google in 2006, and available from `https://catalog.ldc.upenn.edu/LDC2006T13`.

The intuitions behind this approach and the one adopted in Section 4.2.3 are very similar, although formally the latter is based on a vector space model, not on substitution. The approach adopted in Section 4.2.3 is also arguably more complex, because the problem it addresses differs from the problem addressed by [GG08] (and [TM08]) on several aspects. As a reminder, the problem addressed by [GG08] and [TM08] is the following one: given an individual e and a set of concepts Con, identify the concept $C \in Con$ that e is most likely to be an instance of. This is equivalent to deciding for each pair $C_1, C_2 \in Con$ whether $C_1(e)$ is more likely to hold than $C_2(e)$. But the problem addressed by Section 4.2.3 cannot be reduced to that one. The input is a set $\Psi_K = \{C_1(e_1), .., C_n(e_n)\}$ of formulas, and a confidence score (called a plausibility score) needs to be attributed to each of them independently, assuming all other ones are valid, i.e. the training set for each $\psi \in \Psi_K$ is potentially $\Psi_K \setminus \{\psi\}$. In addition, the respective scores of two formulas $C_1(e_1), C_n(e_2) \in \Psi_K$ should be comparable, even when $e_1 \neq e_2$, which was not needed for [GG08] and [TM08], who focused on the case where $C_1 \neq C_2$ but $e_1 = e_2$. In order to answer these requirements, the plausibility score defined in Section 4.2.3 accounts for a potential bias, caused by the fact that the linguistic representation of e_1 or e_2 may be more or less central in the vector space among the representations of all other individuals of $N_{ind}(\Psi_K)$. For instance, if e_1 is more central than e_2, then a high similarity between e_1 and known instances of C_1 is less meaningful than an equally high similarity between e_2 and known instances of C_1. Another specific property of the plausibility score defined in Section 4.2.3 is that it takes into account the possibility that the number of known instances of each concept $C \in N_{Con}(\Psi_K)$ may vary, or in other words, it is (in a certain sense) robust to disproportions in training sets sizes. From some aspects, the task addressed by Section 4.2.3 can also be viewed as more generic than the one addressed by [GG08] and [TM08], in that the concepts of $N_{Con}(\Psi_K)$ are

not assumed to be necessarily disjoint, and some of them may (to a certain extent) be complex DL concepts, not only atomic ones, as discussed in Section 4.2.4.1.

4.2 Proposal

This section presents an original framework in order to compute meaningful empirical linguistic evidence to be integrated to a KB debugging process. The objective is to identify potentially faulty (sets of) axioms within an input KB K, or equivalently some optimal output subbase(s) of K. The approach is based on the assumption that individuals with similar linguistic behaviors tend to instantiate the same concepts in a KB, which is also the intuition underlying the works of [TM08] and [GG08], as just seen in Section 4.1.4.

In order to simplify the presentation, it is assumed throughout the current section that some semantic similarity measure is available, which estimates to what extent the linguistic behaviors of two linguistic terms are similar. Concrete proposals to compute such a similarity measure will be made in Section 4.3, based on distributional semantics. But the framework introduced in the current section is more generic, which is why it temporarily abstracts from these concrete distributional parameters.

Section 4.2.1 justifies the approach, based on the review of ontology learning and population techniques provided in Section 4.1. Section 4.2.3 defines a score which, for each candidate output base Q of the debugging process, and for each consequence ψ of Q of the form $A(e)$ or $\neg A(e)$, with A an atomic DL concept and e an individual, evaluates to what extent ψ is likely to hold if the rest of Q does. Section 4.2.4 discusses possible extensions of this approach, in particular to more complex DL concepts. Then Section 4.2.5 defines different orderings of the candidate subbases based on these scores.

The computation of the set of candidate subbases and the concrete algorithms

used to select optimal subbases are not the subject of this chapter, and will be addressed instead in chapters 6 and 8.

4.2.1 Justification of the approach

4.2.1.1 Linguistic grounding of the input KB

For a collection of texts to confirm or infirm some information expressed by a KB, a trace of the signature of the KB must be found in it. The elements of the signature of an OWL KB generally have one or several associated linguistic labels, which are natural candidates for this. For instance, the linguistic terms "Woody Allen" and "W. Allen" may be two labels associated to the individual *Woody Allen*, and the term "person" may be a label associated to the atomic concept `Person`.

For an atomic concept c though, it is very frequent that the usage of its label in natural language texts does not reflect the meaning of c in the KB. There are at least two reasons for this. The first one is that the signature of a KB often contains ad-hoc atomic concepts, which are relevant in a given applicative scenario, but were not conceived with linguistic usage in mind. A simple example is the Government ontology published as part of the egov project.[5] It contains atomic concepts with labels such as "referred to committee event", "constraint violation level" or "structuring event type". These labels are perfectly meaningful within the KB, but do not correspond to terms actually used in natural language texts (the number of web pages indexed by Google and containing these terms is 6, 342 and 3 respectively). The second reason is that labels associated to concepts in OWL KB are usually common nouns or common noun phrases, and as such tend to be relatively

[5]The precise ontology referenced here is named "oe1gov", and can be downloaded from the url `http://oegov.org/`.

polysemous. In particular, it is very common for a KB's signature to contain atomic concepts which have a clear and specific meaning in that KB, but are labeled with an extremely polysemous linguistic term, such as "function", "model", "rule", "area", "list", "group", "role", "branch", "element", etc.

This is even worse for the labels of DL roles (binary predicates). Consider for instance the label "director" for the DL role `director` in DBpedia. First, it is meant as a relation between audiovisual works and Film directors, which is not the only possible usage of the word "director". But the order of the arguments is also ambiguous (here it should be understood as "was directed by", with the director as the second argument, and not the inverse). Finally, the sequence of words "$< X_1 >$ director $< X_2 >$", with $< X_1 >$ denoting a movie and $< X_2 >$ its director (or the inverse), is very unlikely to be found in a text, such that there is no immediate correspondence between the label of this relation and the way it may be expressed in natural language. All three remarks are also valid for the following DBpedia role labels: "distributor", "editing", "field", "format", "key person", ...

This is why the choice was made in this thesis to focus on the labels of individuals. These labels are prototypically proper names, such a "Woody Allen" or "Egypt", (although this is not necessarily the case, a counterexample being "C.E.O", in Chapter 1, example 1.1.1), and as such, are generally less ambiguous than the common nouns or common noun phrases labeling atomic concepts.

The problem of poylsemy is obviously not completely addressed by this choice though. In particular, homonymy can be an issue, for instance the label "JFK", which may designate a politician or an airport. This is why additional precautions where taken in order to rule out labels with potential homonyms for the experiments described in chapters 6 and 8. These heuristics are described in Chapter 5 Section 5.3.1, together with the constitution of the datasets.

4.2.1.2 Single occurrences

The second choice has already been discussed in Section 4.1.1, with the review of relation extraction techniques based on cooccurrence patterns. In a debugging scenario, the signature sig(K) of the input KB K is known in advance, as opposed to relation extraction, where fresh instances of a relation are to be retrieved. But finding a statistically meaningful number of cooccurrences of pairs of labels of elements of sig(K) is very unlikely, even with the whole web available Therefore the choice was made to focus on single occurrences of target labels, and not on cooccurrences.

4.2.1.3 Distributional evidence

The last choice was guided by the empirical results from [TM08] and [GG08], which indicate that individuals whose labels have a strong distributional similarity tend to instantiate the same atomic concepts, and conversely. This corroborates the intuition that selectional preferences (introduced in Section 4.1.3) are not limited to very abstract semantic types, but may also apply to finer-grained concepts.

Therefore a unique assumption dictates in what follows the use of linguistic evidence for KB debugging, which can be intuitively formulated as follows: a KB is more in line with a corpus if the individuals which instantiate the same concepts according to this KB also tend to have similar linguistic behavior in natural language texts.

In order to avoid possible misunderstandings, it must be emphasized that the whole KB cn be evaluated based on this strategy, including it TBox, as illustrated by the running example given below in Section 4.2.2.

Another important remark is that the framework introduced here is not limited to the similarity between instances of the same *atomic* DL concepts, but may be

extended in some cases to more complex DL concepts (as a reminder of Chapter 2 Section 2.3.2, arbitrarily complex concepts can be built inductively in most DLs). In practice though, some limit needs to be put on the complexity of these concepts, and selectional preferences may not be relevant for the more complex ones. In order to keep the presentation simple, Section 4.2.3 focuses on atomic concepts (e.g. `Person`) and their negation (e.g. $\neg$`Person`), and the generalization to more complex DL concepts is then discussed in Section 4.2.4.1.

4.2.2 Running example

The following set of DBpedia statements will be used as a running example throughout the rest of this section:

Ex 4.2.1 $\Omega = \{$

(1) `owningCompany`(*Smithsonian Networks*, *Smithsonian Institution*),

(2) `doctoralAdvisor`(*Thaddeus S.C. Lowe*, *Smithsonian Institution*),

(3) `doctoralAdvisor`(*Nick Katz*, *Bernard Dwork*),

(4) $\top \sqsubseteq \forall$`doctoralAdvisor.Person`,

(5) $\top \sqsubseteq \forall$`owningCompany.Company`$\}$

From (1), (2), (4) and (5), the individual *Smithsonian Institution* must be an instance of both `Company` and `Person`, which may seem counterintuitive, and indeed does not correspond to the overall understanding of these two concepts within DBpedia.

Let us assume that Ω is part of a larger and consistent KB K, for instance a subset of DBpedia extracted for a specific application, or a set of OWL statements aggregated from multiple sources. Let us assume also that there are several

other instances of `Person` and `Company` according to K and, to keep the example simple, that *Smithsonian Institution*, *Bernard Dwork*, `doctoralAdvisor`, and `owningCompany` do not appear in $K \setminus \Omega$. If most instances of `Person` and `Company` according to K are respectively human beings and companies, one can expect the term "the Smithsonian Institution" to appear with linguistic contexts which tend to characterize terms denoting other instances of `Company` according to K (e.g. the context "X was established"), but less often with contexts which tend to characterize other instances of `Person` (like "X was born in"). Similarly, "Bernard Dwork" should appear with contexts which are characteristic of terms denoting other instances of `Person` according to K. In other words, by checking the overall compliance of K with some linguistic input, it should be possible to identify some undesirable (`Person`(*Smithsonian Institution*)) and desirable (`Company`(*Smithsonian Institution*), `Person`(*Bernard Dwork*)) consequences of it. Another way of viewing this is that some subbases of K should be more plausible than others given the linguistic input. For instance, let $\psi_1 =$ `Company`(*Smithsonian Institution*), and $\psi_2 =$ `Person`(*Smithsonian Institution*). And let $Q_1 = K \setminus \{(1)\}$, $Q_2 = K \setminus \{(2)\}$, etc. Then $Q_1 \nvdash \psi_1$, and $Q_1 \vdash \psi_2$, whereas $Q_2 \vdash \psi_1$, and $Q_2 \nvdash \psi_2$, such that *ceteris paribus*, distributional evidence should favor Q_2 over Q_1.

This holds if TBox axioms are discarded as well. Or in other words, even if the gathered evidence is based on individuals, it may still allow for the evaluation of the removal of TBox axioms. For instance, just as for Q_2, it holds that $Q_4 \nvdash \psi_2$, which confirms the hypothesis that Q_4 is a good candidate output. But it also holds that $Q_4 \nvdash$ `Person`(*Bernard Dwork*), such that distributional evidence should still favor Q_2 over Q_4, but Q_4 over Q_1.

4.2.3 Plausibility

This section defines a score called a *plausibility* score which, given a consistent set Q of axioms and a consequence ψ of Q of the form $A(e)$ or $\neg A(e)$, with e an individual and A an atomic DL concept, evaluates to what extent ψ is likely to hold if the rest of Q does, or equivalently to what extent ψ can be viewed as an "outlier" among consequences of Q.

In what follows, $\text{inst}_Q(A)$ will designate all instances of A according to Q for which a distributional representation could be computed. Similarly, $\text{inst}_Q(\top)$ will designate all individuals in $N_{Ind}(Q)$ for which a distributional representation could be computed. Then Ψ_Q will designate all consequences of Q of the form $A(e)$ or $\neg A(e)$ such that $e \in \text{inst}_Q(\top)$, with A a DL atomic concept and e an individual.

If $\psi \in \Psi_Q$, the set $\sup_Q(\psi)$ will be called the *support set* for ψ in Q. If $\psi = A(e)$, with A an atomic DL concept and e an individual, then $\sup_Q(A(e))$ is defined by:

Definition 4.2.3.1. Support set for $A(e)$
$\sup_Q(A(e)) = \text{inst}_Q(A) \setminus \{e\}$.

Intuitively, $\sup_Q(A(e))$ designates the individuals to which e will be compared, in order to determine whether of not $A(e)$ is likely to hold. If $\sup_Q(A(e)) = \emptyset$, or if $|\sup_Q(A(e))|$ is below a given threshold, then no plausibility score is computed for $A(e)$.

Let $\text{sim}(e_1, e_2)$ be a measure of similarity between the distributional representations of the linguistic labels of individuals e_1 and e_2. Then for each $e' \in \sup_Q(A(e))$, if $\text{sim}(e, e')$ is lower than what could be expected if e' was a random individual of $\text{inst}_Q(\top) \setminus \{e\}$ (i.e. not necessarily an instance of A), the hypothesis that $A(e)$ is an outlier among consequences of Q will be reinforced.

For instance, in example 4.2.1 Section 4.2.2, let $\psi = \texttt{Person}(\mathit{Smithsonian\ Institution})$ and $Q = K$. Then the support set $\sup_K(\psi)$ for ψ in K is composed of all other instances of `Person` according to K. For each individual $e' \in \sup_K(\psi)$, if the similarity $\mathrm{sim}(\mathit{Smithsonian\ Institution}, e')$ is lower than what can be expected for a random individual of $\mathrm{inst}_K(\top) \setminus \{\mathit{Smithsonian\ Institution}\}$, then the confidence in ψ should decline. Conversely, if $\mathrm{sim}(\mathit{Smithsonian\ Institution}, e')$ is higher than expected for a random individual of $\mathrm{inst}_K(\top) \setminus \{\mathit{Smithsonian\ Institution}\}$, the hypothesis that ψ is in line with $\mathrm{Cn}(K)$ (i.e. with K as a theory) will be reinforced.

Here is a cost-efficient method to compute a plausibility score $\mathrm{sc}_Q(\psi)$ for a formula $\psi \in \Psi_Q$ of the form $A(e)$. If $\sup_Q(A(e))$ is the support set for $A(e)$ in Q, then $|\sup_Q(A(e))|$ is the cardinality of $\sup_Q(A(e))$, i.e. the number of other instances of A according to Q. Now let us assume a set W of $|\sup_Q(A(e))|$ randomly chosen individuals in $\mathrm{inst}_Q(\top) \setminus \{e\}$, i.e. of $|\sup_Q(A(e))|$ individuals which are different from e, but not necessarily instances of A. And let the random variable $X^Q_{e,|\sup_Q(A(e))|}$ model the expected value of $\sum_{e' \in W} \frac{\mathrm{sim}(e,e')}{|W|}$, i.e. the mean of the similarities between e and each individual of W. In other words, if $|\sup_Q(A(e))| = |W|$ individuals were randomly chosen in the signature of Q instead of those of the support set specifically, $X^Q_{e,|\sup_Q(A(e))|}$ models the expected average similarity between e and these individuals. Then the plausibility score $\mathrm{sc}_Q(A(e))$ of $A(e)$ can be defined as follows:

Definition 4.2.3.2. Plausibility of $A(e)$

$$\mathrm{sc}_Q(A(e)) = p(X^Q_{e,|\sup_Q(A(e))|} \leq \sum_{e' \in \sup_Q(A(e))} \frac{\mathrm{sim}(e,e')}{|\sup_Q(A(e))|})$$

$\mathrm{sc}_Q(A(e))$ estimates of how surprisingly high the similarity between e and the individuals of the support set $\sup_Q(A(e))$ is, considering the overall similarity between e and the individuals of Q.

A first interesting property of this score is that it accounts for the fact that the

distributional representation of e may be more or less central among $\text{inst}_Q(\top)$. For instance, let $e_1 \in \text{inst}_Q(A_1)$ and $e_2 \in \text{inst}_Q(A_2)$, such that the average similarity between e_1 and other instances of A_1 is identical to the average similarity between e_2 and other instances of A_2, i.e. $\sum_{e' \in \text{sup}_Q(A_1(e_1))} \frac{\text{sim}(e_1,e')}{|\text{sup}_Q(A(e_1))|} = \sum_{e' \in \text{sup}_Q(A_2(e_2))} \frac{\text{sim}(e_2,e')}{|\text{sup}_Q(A_2(e_2))|}$. It may still be the case that e_1 is more similar on average to a random individual from $\text{inst}_Q(\top) \setminus \{e_1\}$ than to another instance of A_1, whereas the contrary holds for e_2 and A_2. In this case, one would like to obtain $\text{sc}_Q(A_1(e_1)) < 0.5$ and $\text{sc}_Q(A_2(e_2)) > 0.5$, which is guaranteed by the above definition.

Another desirable property of such a score is that it accounts for the fact that the sizes of the support sets for two consequences may vary. For instance, let us assume that two consequences $A_1(e)$ and $A_2(e)$ are evaluated, and that there are 20 other occurrences of A_1 according to Q, against 200 other occurrences of A_2. And let us assume also that the average similarity between e and other instances of A_1 is equal to the average similarity between e and other instances of A_2, i.e. $\sum_{e' \in \text{sup}_Q(A_1(e))} \frac{\text{sim}(e,e')}{|\text{sup}_Q(A_1(e))|} = \sum_{e' \in \text{sup}_Q(A_2(e))} \frac{\text{sim}(e,e')}{|\text{sup}_Q(A_2(e))|}$. Then this value should be considered more informative for $A_2(e)$ than for $A_1(e)$, because there is more evidence supporting it. This intuition can simply be captured by the modeling of the random variable $X^Q_{e,|\text{sup}_Q(A_i(e))|}$. For the experiments described in Chapters 6, and 8, $X^Q_{e,|\text{sup}_Q(A_i(e))|}$ was assumed to follow a beta distribution $\text{Beta}(\alpha, \beta)$, which allows taking the size $|\text{sup}_Q(A_i(e))|$ of the support set into account, where a normal distribution for instance would not. The lower $|\text{sup}_Q(A_i(e))|$ is, the more uniform the distribution of $X^Q_{e,|\text{sup}_Q(A_i(e))|}$ should be. This can be obtained by setting $X^Q_{e,|\text{sup}_Q(A_i(e))|} \sim \text{Beta}(m \cdot |\text{sup}_Q(A_i(e))| + 1, (1-m) \cdot |\text{sup}_Q(A_i(e))| + 1)$, where m is the average similarity between e and all other individuals of $\text{inst}_Q(\top)$, i.e. $m = \sum_{e' \in \text{inst}_Q(\top) \setminus \{e\}} \frac{\text{sim}(e,e')}{|\text{inst}_Q(\top)|-1}$.

A possible interrogation here is the choice of $\text{inst}_Q(A) \setminus \{e\}$ as the support set for $\psi = A(e)$. For instance, if $\psi = \texttt{Person}(\textit{Bernard Dwork})$, a case could be made for using $\text{inst}_Q(\neg A)$ as well, i.e. for exploiting the similarity (or dissimilarity) between *Bernard Dwork* and individuals which, according to K, are instances of $\neg\texttt{Person}$.[6] This is a very unrealistic assumption though from a linguistic point of view, which can simply be seen here by replacing `Person` with `JazzComposer` for instance. Let us assume that *Thelonious Monk* and *Beijing* are reliable instances of `JazzComposer` and $\neg$`JazzComposer` respectively, and that *Bernard Dwork* should not be considered as an instance of `JazzComposer`. This is clearly not sufficient to expect that $\text{sim}(\textit{Bernard Dwork}, \textit{Beijing}) > \text{sim}(\textit{Bernard Dwork}, \textit{Thelonious Monk})$. In other words, the fact that two individuals are both non-instances of a same atomic concept is not sufficient to assume that their respective labels should have similar linguistic behaviors.

Interestingly enough, and for the same reason, the support set for a consequence $\psi \in \Psi_Q$ of the form $\neg A(e)$ is not $\text{inst}_Q(\neg A) \setminus \{e\}$, but:

Definition 4.2.3.3. Support set for $\neg A(e)$
$\text{sup}_Q(\neg A(e)) = \text{inst}_Q(A)$

And the plausibility for $\neg A(e)$ should this time evaluate how surprisingly *low* the similarity between e and the individuals of the support set $\text{sup}_Q(\neg A(e))$ is, considering the overall similarity between e and the individuals of Q, i.e.:

Definition 4.2.3.4. Plausibility for $\neg A(e)$
$\text{sc}_Q(\neg A(e)) = p(X^Q_{e,|\text{sup}_Q(\neg A(e))|} \geq \sum_{e' \in \text{sup}_Q(\neg A(e))} \frac{\text{sim}(e,e')}{|\text{sup}_Q(\neg A(e))|})$

[6] i.e. such that $Q \vdash \neg\texttt{Person}(e')$ and not only such that $Q \nvdash \texttt{Person}(e')$

4.2.4 Extensions

4.2.4.1 Complex concepts

A first possible possible extension of the above plausibility score concerns consequences of the form $C(e)$, where C is an arbitrary DL concept, and not necessary an atomic concept or its negation.

For instance, in example 4.2.1 above, let $Q = K$, $C_1 = \exists \texttt{doctoralAdvisor}.\top$ and $C_2 = \exists \texttt{doctoralAdvisor}^-.\top$. Then the set Ψ_Q of evaluated consequences of Q may be extended "for free" with $C_1(\textit{Nick Katz})$, $C_1(\textit{Thaddeus S.C. Lowe})$, $C_2(\textit{Bernard Dwork})$ and $C_2(\textit{Smithsonian Institution})$. But also further on with consequences such as:

$\exists \texttt{doctoralAdvisor.Company}(\textit{Thaddeus S.C. Lowe})$,

$\forall \texttt{owningCompany.Company}(\textit{Smithsonian Networks})$,

$\forall \texttt{doctoralAdvisor.Person}(\textit{Smithsonian Networks})$,

$\exists \texttt{doctoralAdvisor.Company} \sqcap \texttt{Person}(\textit{Thaddeus S.C. Lowe})$,

$\exists \texttt{owningCompany} \exists \texttt{doctoralAdvisor}^-.\texttt{Person}(\textit{Smithsonian Networks})$, ...

As explained in Chapter 2 Section 2.3.2, in most DLs, the set of concepts which can be inductively built out of a given signature may be infinite, such that some limit to this concept construction mechanism is required. Unfortunately, there is no obvious semantic solution to this problem. In many DLs, if Q is a set of axioms and Ψ_Q^+ the set of all consequences of Q of the form $C(e)$, with C an arbitrarily complex DL concept, then there may be no finite subset Δ of Ψ_Q^+ such that $\Delta \vdash \psi$ for all $\psi \in \Psi_Q^+$. For instance, let $Q = \{\top \sqsubseteq \forall R.A, \top(e_1)\}$, Then for any finite $\Delta \subset \Psi_Q^+$, there is a formula $\psi = \forall R.\forall R. \ldots \forall R.A(e_1) \in \mathrm{Cn}(Q) \setminus \mathrm{Cn}(\Delta)$.

Some complex concepts also seem intuitively more relevant than others from a

linguistic point of view. For instance, it is arguably doubtful that instances of the concept of "people whose father lives in an apartment" present similar linguistic behavior for that reason. On the other hand, concepts like "things owned by someone" or "things created by someone" seem like plausible candidates.

The upcoming experiments focus on atomic concepts and their negation. But an interesting continuation of this work would be to extend this set of concepts with concepts of the form $\exists R.\top$ and $\exists R^{-}.\top$ (where R is an atomic DL role), as well as their respective negations $\forall R.\bot$ and $\forall R^{-}.\bot$. So for instance, if $\texttt{own}$ is a DL role in the signature of Q, then for each $e \in \text{inst}_Q(\exists \texttt{own}.\top)$, the plausibility of $\exists \texttt{own}.\top(e)$ can be evaluated, i.e. intuitively to what extent "e owns something" is likely to hold, and similarly for the DL concepts $\exists \texttt{own}^{-}.\top$ (i.e. individuals "owned by something/someone" according to Q), $\forall \texttt{own}.\bot$ (i.e. individuals "who/which cannot own") and $\forall \texttt{own}^{-}.\bot$ (i.e. individuals "who/which cannot be owned").

4.2.4.2 Distinct individuals

Another possible variation of this framework, not experimented in this work though, consists in considering consequences of Q of the form $e_1 \neq e_2$ (i.e. the fact that e_1 and e_2 are not the same individual) instead of $C(e)$. The unique name assumption is not made in OWL, which means that two distinct individuals may be interpreted identically, and therefore these consequences do not hold by default. They may be explicitly stated in Q, but are more often only entailed by Q, provided it contains some form of negation.

So if Q is a candidate output bases, and if $e_1, e_2 \in \text{inst}_Q(\top)$ such that the similarity between e_1 and e_2 is higher than what one may expect for two random individuals of $\text{sig}(Q)$, then the confidence in Q will decrease if $Q \vdash e_1 \neq e_2$ (and/or possibly

increase if $Q \nvdash e_1 \neq e_2$). Conversely, if the similarity between e_1 and e_2 is lower than what one may expect for two random individuals of $\mathrm{sig}(Q)$, then the confidence in Q will increase if $Q \vdash e_1 \neq e_2$ (and/or possibly decrease if $Q \nvdash e_1 \neq e_2$).

4.2.5 Ranking candidate subbases

As already illustrated by example 4.2.1, it is assumed throughout this chapter that the objective of the debugging process is to select some optimal subbase(s) of a consistent input KB K, from a predefined family $\mathcal{Q} \subseteq 2^K$ of candidate subbases.

If $Q \in \mathcal{Q}$, Then Ψ_Q will designate the set of all consequences of Q for which a linguistic plausibility score could be computed. As a reminder, these consequences are of the form $C(e)$, with C a DL concept and e an individual (for more details on the form of C, see section 4.2.4.1). Intuitively, for each $\psi \in \Psi_Q$, and *ceteris paribus*, the higher the plausibility score $\mathrm{sc}_Q(\psi)$, the more likely it is for Q to be a good candidate within $\mathcal{Q}$, as illustrated by example 4.2.1.

This section investigates how these plausibility scores can be aggregated in order to compute a preference relation over $\mathcal{Q}$.

A first straightforward option consists in computing a score $\mathrm{comp}(Q)$ (for "compliance") for each $Q \in \mathcal{Q}$, defined as the mean of the plausibility scores obtained for each $\psi \in \Psi_Q$, i.e.:

Definition 4.2.5.1. $\mathrm{comp}(Q) = \sum_{\psi \in \Psi_Q} \frac{\mathrm{sc}_Q(\psi)}{|\Psi_Q|}$

A total preorder $\preceq_{\mathrm{comp}}$ can then simply be defined over $\mathcal{Q}$ based on these scores as follows: [7]

[7] The assumption is made that a minimum of syntactic information should be lost whenever possible. Redundancies in this view should be preserved whenever possible, e.g. if $\mathrm{Cn}(Q_1) = \mathrm{Cn}(Q_2)$ and $Q_1 \subset Q_2$, then $Q_1 \prec_{\mathrm{comp}} Q_2$ still holds.

Definition 4.2.5.2. $\preceq_{\text{comp}}$

- $Q_1 \preceq_{\text{comp}} Q_2$ iff $Q_2 \not\prec_{\text{comp}} Q_1$
- $Q_1 \prec_{\text{comp}} Q_2$ iff either $\text{comp}(Q_1) < \text{comp}(Q_2)$, or ($\text{comp}(Q_1) = \text{comp}(Q_2)$ and $Q_1 \subset Q_2$))

Then a subbase Q of K can be viewed as optimal iff it is maximal wrt $\preceq_{\text{comp}}$.[8] As will be discussed in Chapter 6 Section 6.3.1 though, identifying optimal subbases is a non trivial task, even for a moderately large $\mathcal{Q}$, one of the reasons being that a same consequence ψ may have different plausibility scores wrt two candidate bases in $\mathcal{Q}$.

As an alternative to the function comp(), and in order to avoid the fact that a same consequence may have different plausibility scores wrt two subbases of K, one may choose to discard unlikely consequences based on their respective scores wrt K, i.e. to use the score $\text{comp}_K(Q)$, defined by:

Definition 4.2.5.3. $\text{comp}_K(Q) = \sum_{\psi \in \Psi_Q} \frac{\text{sc}_K(\psi)}{|\Psi_Q|}$

The corresponding preference relation $\preceq_{\text{comp}_K}$ over $\mathcal{Q}$ is defined as previously:

Definition 4.2.5.4. $\preceq_{\text{comp}_K}$

- $Q_1 \preceq_{\text{comp}_K} Q_2$ iff $Q_2 \not\prec_{\text{comp}_K} Q_1$
- $Q_1 \prec_{\text{comp}_K} Q_2$ iff either $\text{comp}_K(Q_1) < \text{comp}_K(Q_2)$, or ($\text{comp}_K(Q_1) = \text{comp}_K(Q_2)$ and $Q_1 \subset Q_2$))

This solution is arguably less satisfying, because it relies on the plausibility score $\text{sc}_K(\psi)$ (in the definition of $\text{comp}_K(Q)$), and not on $\text{sc}_Q(\psi)$. For instance, if $\psi = A(e)$,

[8] There may be several several optimal subbases.

the support set (see section 4.2.3) used to compute $\mathrm{sc}_K(\psi)$ is the set of other instances of A according to K, but there may be less of them according to Q, such that $\mathrm{sc}_K(\psi)$ may not accurately reflect whether ψ is likely to hold if the rest of Ψ_Q does, but only if the rest of Ψ_K does. This solution is more amenable to optimizations though, as will be discussed in Chapter 6 Section 6.3.1.

Alternatively, instead of taking the mean of the scores of evaluated consequences of Q, one may want to penalize the candidate subbases with the most unlikely consequences, which gives a standard (total) lexicographic ordering $\preceq_l$ over $\mathcal{Q}$, defined as follows. For a given candidate base Q, let $(\psi_1, .., \psi_{|\Psi_Q|})$ designate the formulas of Ψ_Q ordered by increasing plausibility score, and let $\boldsymbol{w}_Q = (w_Q^1, .., w_Q^{|\Psi_Q|})$ be the vector defined by $w_Q^i = \mathrm{sc}_Q(\psi_i)$ for all $1 \leq i \leq |\Psi_Q|$. For instance, if $\Psi_Q = \{\psi_1, \psi_2, \psi_3\}$, and if $\mathrm{sc}_Q(\psi_1) = 0.2$, $\mathrm{sc}_Q(\psi_2) = 0.5$ and $\mathrm{sc}_Q(\psi_3) = 0.2$, then $\boldsymbol{w}_Q = (0.2, 0.2, 0.5)$. Let $\preceq_l$ be a standard ascending lexicographic ordering of all $\boldsymbol{w}_Q$, i.e. $\boldsymbol{w}_{Q_1} \preceq_l \boldsymbol{w}_{Q_2}$ iff $\boldsymbol{w}_{Q_2} \not\prec_l \boldsymbol{w}_{Q_1}$, and $\boldsymbol{w}_{Q_1} \prec_l \boldsymbol{w}_{Q_2}$ iff there is a $1 \leq i \leq |\Psi_{Q_2}|$ such that $w_{Q_1}^j = w_{Q_2}^j$ for all $1 \leq j < i$, and either $w_{Q_1}^i < w_{Q_2}^i$ or $|\Psi_{Q_1}| = i - 1$. For instance, if $\boldsymbol{w}_{Q_1} = (0.2, 0.2, 0.5, 0.1)$ and $\boldsymbol{w}_{Q_2} = (0.2, 0.3, 0.5)$, then $\boldsymbol{w}_{Q_1} \prec_l \boldsymbol{w}_{Q_2}$, because $w_{Q_1}^1 = w_{Q_2}^1 = 0.2$, and $w_{Q_1}^2 = 0.2 < 0.3 = w_{Q_2}^2$.

Then a preference relation $\preceq_{lex}$ over $\mathcal{Q}$ can be defined almost as previously:

Definition 4.2.5.5. $\preceq_{lex}$

- $Q_1 \preceq_{lex} Q_2$ iff $Q_2 \not\prec_{lex} Q_1$
- $Q_1 \prec_{lex} Q_2$ iff either $\boldsymbol{w}_{Q_1} \prec_l \boldsymbol{w}_{Q_2}$, or ($\boldsymbol{w}_{Q_1} =_l \boldsymbol{w}_{Q_2}$ and $Q_1 \subset Q_2$)

Again, $\mathrm{sc}_K(\psi)$ may be used instead of $\mathrm{sc}_Q(\psi)$, yielding the lexical ordering $\preceq_{l_K}$, and the preference relation $\preceq_{lex_K}$ over $\mathcal{Q}$, defined by:

Definition 4.2.5.6. $\preceq_{lex_K}$

- $Q_1 \preceq_{lex_K} Q_2$ iff $Q_2 \prec_{lex_K} Q_1$
- $Q_1 \prec_{lex_K} Q_2$ iff either $\boldsymbol{w}_{Q_1} \prec_{l_K} \boldsymbol{w}_{Q_2}$, or ($\boldsymbol{w}_{Q_1} =_{l_K} \boldsymbol{w}_{Q_2}$ and $Q_1 \subset Q_2$)

This last possibility corresponds to a relatively intuitive operation, which consists in giving up in priority the consequences of K which are the most implausible wrt K.

All four possibilities will be used in the following chapters.

4.3 Distributional similarity

This section is an introduction to the computation of distributional similarity between words, together with a presentation of the specific distributional settings used for the experiments presented in chapters 6 and 8.

The introduction is partly based on [TPo10], with a focus on the "word-context" class of distributional approaches according to the terminology of the authors. It remains very simple on purpose, because the distributional techniques used for the work presented here are relatively standard. In particular, many advances in distributional semantics deal in some way with dimensionality reduction, which was simply not needed here, as explained in Section 4.3.1.

Throughout this section, "word" will generally be used as a shortcut for "word or multi-word unit", and "target words" will designate the words (or multi-word units) for which a distributional representation is needed. In particular, a target word may be a term labeling an individual, like "'Woody Allen" or "the Smithsonian Institution". So a "words by contexts" matrix of is a shortcut for a "words or multi-word units by contexts" matrix, and each row vector in such a matrix is the distributional representation of a target word.

4.3.1 Vector space model

Distributional semantics manipulate the meaning of words through algebraic structures which represent the linguistic contexts in which these words are observed, sometimes called their *distribution*. In its simplest form, the distributional representation of the meaning of some target words in a given corpus is a matrix $\mathbf{X}$ of m target words by n linguistic contexts, where each row vector $\mathbf{x}_{i:}$ corresponds to the distribution of a target word, each column vector $\mathbf{x}_{:j}$ corresponds to the distribution of a context, and each value x_{ij} is based on the frequency of target word i with context j in the corpus. Alternatively, the target word by context matrix is often viewed as an n-dimensional vector space, and each target word as a vector in this space.

Corpora used in distributional semantics for general linguistic purposes tend to be large (up to billions of words), and the number of target words is generally important. For instance, the target words may be all common nouns appearing in the corpus (or all common nouns above a certain frequency threshold), all adjectives, etc. Therefore the number n of different contexts observed with these target words (or equivalently the number of dimensions of the vector space) may be very important (depending on the adopted representation of contexts, discussed in Section 4.3.3), and the matrix extremely sparse.

A variety of dimensionality reduction strategies address this sparsity issue, one of the most popular being singular value decomposition (SVD), which amounts to computing two matrices $\mathbf{M}$ and $\mathbf{N}$ of dimensions $m \times t$ and $t \times n$ respectively, with $t \ll n$, such that their product $\mathbf{MN}$ is an optimal approximation of $\mathbf{X}$ given t. Similarity between target words can then be computed based on their respective vectors in $\mathbf{M}$, instead of their respective vectors in $\mathbf{X}$. More recent advances in the

field are based on the popular deep learning paradigm, which also produces low-dimensional distributional representations of words, but intuitively as a side-effect of the optimization of a neural network for some independent classification task. The respective (non-strictly computational) benefits of the different dimensionality reduction techniques used in NLP are still debated upon, and seem to vary with the application.

By contrast, the debugging strategy developed here focuses on the distributions of a relatively small number of target words, which are the labels of the individuals of the input KB. For the experiments described in chapters 6 and 8, this represents a few thousand target words at most, and a few hundred occurrences of each of them are needed only. Therefore, after elimination of contexts which appear only with one target word, the word by context matrix $\mathbf{X}$ is of a very manageable size, which is why its structure has been left untouched by default. But if the debugging approaches presented in this thesis are to be applied to (much) larger input KBs, then a dimensionality reduction will probably be needed. The choice of the most appropriate dimensionality reduction technique remains an open question though, and may need to be made on an empirical basis.

Vectors are not the only type of algebraic structures used to represent the meaning of words in contexts, and neither is deriving a similarity measure the only possible usage of such representations. Recent advances in the field focus in particular on the problem of compositionality (see [Cla13] for an introduction), relying on higher order tensors, but are clearly out of the scope of this work.

4.3.2 Corpus

As already emphasized in Section 4.1, a particularity of KB debugging, as opposed to ontology learning for instance, is that the signature of the input KB K is known in advance.

Distributional similarity is used here to evaluate whether or not two given target words from a predefined target words list (the labels denoting individuals of $\text{sig}(K)$) behave similarly. Therefore, for the similarity scores to be meaningful, the corpus must contain a sufficient number of occurrences of these target words, such that some target words actually share linguistic contexts.

It is extremely unlikely though to find a sufficient number of occurrences of these terms in one of the linguistically processed corpora traditionally used in corpus linguistics, even the larger ones, like the ukWaC corpus [BBFZ09]. For instance, focusing only on the five individuals of example 4.2.1 Section 4.2.2, it can be safely assumed that no existing linguistically processed corpus contains 200 occurrences of each of "the Smithsonian Institution", "Smithsonian Networks", "Thaddeus S.C. Lowe", "Nick Katz" and "Bernard Dwork".

On the other hand, it is relatively easy to retrieve a sufficient number of web pages containing these terms, which is why the choice has been made to build a corpus dynamically for each input KB K, relying a web search engine, with the individual labels as queries. This yields relatively small corpora (up to 25 million words for 1500 individuals), but tailored for the input KB. The precise corpus extraction procedure applied in the following experiments is detailed in Chapter 5, together with the description of the various datasets.

4.3.3 Contexts

The representation of linguistic contexts has a direct incidence on the type of similarity between target words which may be extracted from a corpus. In distributional semantics, what is understood as the context(s) in which a target word appears may vary. Let w be a target word,[9] and $\text{occ}_w^1, .., occ_w^n$ be the different occurrences of w in the corpus. One of the simplest representation of contexts is the bag of word approach, where the contexts of an occurrence occ_w^i are all other words appearing in the whole document which contains occ_t^i, or just the paragraph containing it, or a smaller span, like a sentence or a window of k words surrounding w.

The matrix $\mathbf{X}$ is these cases is a word by word matrix. If the context span is very small, and if the row vectors for two target words w_1 and w_2 are very similar, this tends to indicate that w_1 and w_2 may be replaced by each other in the corpus. But if the context span is large, this high similarity may be due to another phenomenon, namely the fact that w_1 and w_2 tend to occur together in the same documents or paragraphs.

For that reason, in the bag of word approach, small context spans tend to produce similarity measures which reflect lexical relations between target words such as synonymy, antonymy or co-hyponymy (like the similarity between "nurse" and "doctor"), whereas large context spans tend to produce similarity measures which reflect topical similarity between target words (like "nurse" and "hospital"). The first type of similarity is clearly more relevant for the problem at hand here.

But even for small context spans, the bag of word model may seem too coarse-grained to capture the similarity of usage of target words, and more elaborated linguistic representations may be needed. In particular, the bag or words model

[9] Again, "target word" stands here for "target word or multi-word unit".

abstracts from the sequencing of surrounding words, as well as the position of the target word in that sequence. For instance, "$< w_1 >$ grew up in" and "grew up in $< w_2 >$" intuitively do not select the same semantic types (probably a human being in the first case, probably a location in the second), but are identical in a bag of words representation (precisely, 3 contexts are observed with w_i in each occurrence, namely "grew, "up" and "in"). Another example would be "the first take of $< w_1 >$" and "take the first of $< w_2 >$", where this time the position of the target word is identical, but the order of the cooccurring words differs. This is why for lexical purposes, a context is often represented as a sequence of words surrounding a target word, together with the position of the target word in the sequence.

But the converse may hold as well, i.e. one would like in some cases to consider as identical some contexts which differ on surface, prototypically because they are inflections of the same words. For instance, "lives" and "lived" should probably be understood as the same word in the occurrences "lives in $< w_1 >$" and "lived in $< w_2 >$". A lemmatization of the corpus may therefore be useful, representing each surrounding word by a lemma+part-of-speech pair (lemmaPOS), for instance "live$_{}_V$" which designates the verb "to live" (where V stands for "verb"), regardless of its morphological inflection. But more advanced linguistic preprocessings of the corpus are also frequently applied. For instance, if the corpus has been syntactically parsed, the contexts of a target word may be its (lemmatized) syntactic governors or dependents, together with the corresponding syntactic function (subject, modifier, ...). Some categories of less meaningful context words, like determiners, may also be ignored.

Two alternative representations of contexts were adopted for the experiments presented in this thesis, each of them being a different compromise, dictated by practical considerations. The first representation relies on some linguistic preprocessing of the

corpus, namely a lemmaPOS tagging performed with the Stanford Tagger [TKMS03], using a pre-trained model for English. A context is defined as a sequence of 2 to 5 lemmaPOS together with the position of the target term in it, and such that the context does not span over a punctuation mark in the text.[10] Some of the parts-of-speech (POS) produced by the Stanford Tagger were generalized though, because of distinctions which are probably irrelevant here, especially distinctions pertaining to morphological inflections. For instance, no distinction was made between singular and plural common nouns, such that the two corresponding POS were merged. Some less meaningful words were also ignored on purpose when extracting these contexts, based on their lemma and/or POS, in particular most determiners, as well as modal and auxiliary verbs ("be", "have", "can", etc.). Syntactic parsing was not used, partly because of its cost (each input KB requires a new corpus, which would need to be analyzed syntactically), but mostly for reliability reasons. Syntactic parsing models are prototypically trained on collections of edited texts such as newspaper articles, and do not perform well on random web pages, unless the edited content of these pages has been previously isolated. The Bootcat library [BB04] was used to clean HTML documents, as well as some additional heuristics. But this is unfortunately not always sufficient to guarantee that the remaining content is syntactically valid English.

The second representation of contexts experimented in this thesis is a more coarse-grained one. A context in this latter case is just defined as a sequence of 2 to 5 words (also called a (2 to 5)-gram) immediately preceding or following an occurrence of a target term, together with the position of the target term (before or after), and such

[10] If w is the target word, taking into consideration these 4 sizes (2 to 5) and the shifting window (for instance, a context of size 2 may start 2 words before w, 1 word before w, or immediately after w), this yields a maximum of $6 + 5 + 4 + 3 = 18$ observed contexts for a unique occurrence of w.

that the context does not span over a punctuation mark in the text. This solution is arguably less satisfying from a linguistic point of view. But it also presents a practical advantage over the previous one. Given a context c and a target term w, it is possible to weight the observed frequency of c with w based on the estimated probability to encounter the sequence of words c in a natural language text, obtained from a so-called "n-gram language model". This weighting strategy is inspired by [GG08], and is presented in the next section.

4.3.4 Selectional preferences and frequency weighting

The values in a word by context matrix are based on the number of occurrences of each target term with each context in the corpus. But as already mentioned in Section 4.1.3, the fact that two target words occur with a very frequent context, for instance immediately followed by the verb "have", is not very informative.

This also holds for contexts which are less frequent, but may appear with a variety of target words. For instance, if target words are proper names, almost any of them is likely to be immediately preceded by the contexts "talk about", "describe" or "present". On the other hand, a context like "hired" or "interrupted" (or "drive" in the example of Section 4.1.3) seems more selective.

This intuition is generally captured in distributional semantics by some weighting function applied to the values (i.e. the frequencies) in a word by context matrix. According to [TPo10], one of the most successful weighting functions for word similarity in word by context matrices is the Positive Pointwise Mutual Information (PPMI), which is equivalent to the well-known Pointwise Mutual Information (PMI), but where all negative values are set to 0. The PMI intuitively evaluates to what extent the probability $p(w, c)$ of an occurrence of target word w with context c is higher

than what may be expected from their respective probabilities of occurrence, i.e. $\text{PMI}(w, c) = \log \frac{p(w,c)}{p(w)p(c)}$.

These probabilities are generally maximum likelihood estimates computed out of the (non-weighted) frequencies in the word by context matrix $\mathbf{X}$. A possible weakness of this strategy for the problem at hand is that these estimates are not sensitive to the sample size. For a given KB K, $\mathbf{X}$ may be relatively small, thus hindering the reliability of such estimates. In particular the number of rows of $\mathbf{X}$ (i.e. the number of target terms) may be very limited, such that the cumulated number of occurrences of a given context c with any target terms may not be representative of its overall usage in natural language texts.

A first simple solution consists in estimating $p(c)$ based on all occurrences of all contexts of interest in the corpus, and not only on their occurrences with target words. But even in this case, because the size of the corpus for a given KB is relatively limited (usually a few million words), this probability estimate may be biased, especially for relatively unusual contexts.

In order to compensate for this small sample size, and following the proposal made by [GG08], another weighting strategy was also applied. It relies on an external n-gram language model, which evaluates the probability of arbitrary sequences of 1 to n words, usually based on large collections of natural language texts, without linguistic preprocessing (such models are for instance heavily used in speech recognition). Then if $p(c)$ is the resulting estimated probability of context c, [GG08] propose to weight the observed frequency of c with w in the smaller corpus by the *self-information* $\text{self}(c)$, simply defined by $\text{self}(c) = -\log p(c)$.

Both weighting functions will be evaluated in the following experiments, either independently, or combined with each other. In the latter case, the weighted frequency for w and c is just $\text{self}(c) \times \text{PPMI}(w, c)$. Self-information could only be applied in

the case where contexts are represented as n-grams though (and not sequences of lemmaPOS), as explained in Section 4.3.3.

4.3.5 Similarity

The most standard measure used to evaluate the semantic similarity of two target words in a word by context matrix is the cosine similarity between their respective row vectors, i.e. the cosine of the (acute) angle formed by their respective representations in the vector space. This is also the measure used in the following experiments.

The cosine similarity intuitively compensates for the fact that the number of occurrences of each target word in the corpus may vary. If $\mathbf{x}_{i:}$ and $\mathbf{x}_{j:}$ are two row vectors in $\mathbf{X}$, then their cosine similarity is given by $\text{sim}(\mathbf{x}_{i:}, \mathbf{x}_{j:}) = \frac{\mathbf{x}_{i:} \cdot \mathbf{x}_{j:}}{||\mathbf{x}_{i:}|| \times ||\mathbf{x}_{j:}||}$, where "$\cdot$" stands for the dot product, and $||\mathbf{x}||$ for the euclidean norm of $\mathbf{x}$.

Although the term "cosine distance" is often used as a synonym for it, the cosine similarity is not a distance in the mathematical sense, as it does not necessarily verify the triangle inequality.

4.4 Conclusion

This chapter identified a form of linguistic evidence which may be integrated to a KB debugging process, with the only requirements that the input KB's signature contains individuals, and that these individuals in turn have linguistic labels (prototypically proper names).

Given a finite and consistent set Q of formulas (e.g. one of the candidate outputs of a debugging process), the proposal consists in evaluating a specific set Ψ_Q of consequences of Q, in order to estimate to what extent Ψ_Q is compliant with some

automatically gathered collection of web pages. This proposal is partly inspired by recent works in ontology population, and based on distributional semantics, the underlying assumption being that individuals of a KB which instantiate the same concepts also tend to share the same linguistic contexts. In its simplest form, Ψ_Q is defined syntactically (up to equivalence) as the set consequences of Q of the form $A(e)$ or $\neg A(e)$, with A an atomic concept and e an individual. But as discussed in Section 4.2.4.1, the approach may be extended to consequences of the form $C(e)$, where C is a more complex DL concept.

The integration of Natural Language Processing and knowledge engineering has mostly been investigated in the fields of information extraction and ontology learning/population. The problem under investigation here (KB debugging) is of a very different nature though, characterized in particular by the fact that the whole signature of interest is known in advance. Section 4.2 reviews some of the main techniques developed in information extraction and ontology learning/population, explaining why they may or may not be appropriate for KB debugging.

The main contribution of this chapter was made in Section 4.2.3. If $\psi \in \Psi_Q$, a *plausibility* score was defined for ψ wrt Ψ_Q, which intuitively estimates to what extent ψ is likely to hold if the rest of Ψ_Q does. Equivalently, this score estimates to what extent ψ is an outlier within Ψ_Q, such that a low plausibility for ψ is likely to be caused by some violation of common sense within Q, like the ones presented in Chapter 1. Then if $\mathcal{Q}$ is the family of candidate output subbases of the debugging process, Section 4.2.5 proposed different ways to aggregate the plausibility scores of all $\psi \in \Psi_Q$ for each $Q \in \mathcal{Q}$, and defined four alternative preference relations over $\mathcal{Q}$, which will guide some of the experiments described in Chapter 6.

Finally, Section 4.3 introduced the distributional settings used in the experiments described in this thesis, which remain relatively basic, because there was no practical

need for a dimensionality reduction, given the relatively small number of target words, namely the labels of the individuals of K.

No indication has been given though as to how these proposals may be integrated to an actual debugging process, and this will be the main topic of Chapter 6, which also provides an independent evaluation of the plausibility score defined in Section 4.2.3.

Chapter 5

Datasets

This chapter presents the datasets used in the evaluations described in the following chapters. Some datasets are used in several evaluations, which is why they are all introduced in the current chapter. The input KBs are of two very different types. KBs of the first type are sets of logical statements automatically extracted from DBpedia [MJB12]. Such KBs are very likely to contain intuitively nonsensical sets of statements, as illustrated in Chapter 1, and therefore are interesting candidate inputs to evaluate a given debugging strategy. As a reminder, the objective being pursued is the automated identification of axioms to be preferably discarded from the KB (or amended) in order to get rid of the nonsense. But evaluating the performance of a given approach on such inputs is non-trivial. Some human judgment is required as to whether the axioms selected for removal are indeed erroneous, which may not be obvious. As a very simple example, consider the two following axioms, taken from example 1.1.1, in Chapter 1:

`keyPerson`(*Caixa Bank*, *CEO*)

`keyPerson`(*Brookfield Office properties*, *Peter Munk*)

There seem to be two intuitively incompatible meanings of `keyPerson` here, ranging either over persons (like *Peter Munk*), or person functions (like *CEO*). But without further information about which of these two meanings prevails within the input KB, deciding that one of them is the incorrect one may be seen as arbitrary. In other words, the whole base needs to be taken into account in order to determine which axioms are outliers, which may in some cases be complex, even for an expert.

This is why an additional and fully automated evaluation protocol was designed, as a complement of an evaluation based on human judgment. It takes as input a KB of an arguably higher quality, but automatically degraded by generating random axioms out of its signature, the underlying assumption being that such random axioms are very likely to be absurd wrt to the rest of the KB. A debugging strategy is then evaluated based on its ability to automatically identify these randomly generated axioms.

Table 5 contains statistics about the seven KBs presented in this chapter. The five first KBs are DBpedia subsets, and the two other ones served as inputs to the degrading procedure.

Section 5.1 briefly introduces DBpedia, and describes the procedure applied to extract the five DBpedia subsets. Section 5.2 introduces the two other KBs, as well as the degrading algorithm. Finally, Section 5.3 focuses on the automatic construction of linguistic corpora out of web queries for each of these seven KBs.

KB	Axioms	DL	Classes	Object Properties	Datatype Properties	Individuals
K_1^{DBP}	8329	$\mathcal{AL}^{(\mathcal{D})}$	106	100	205	1437
K_2^{DBP}	6148	$\mathcal{AL}^{(\mathcal{D})}$	77	69	116	866
$K_{1.1}^{DBP}$	225	$\mathcal{AL}^{(\mathcal{D})}$	27	17	38	29
$K_{1.2}^{DBP}$	159	$\mathcal{AL}^{(\mathcal{D})}$	16	11	18	28
$K_{1.3}^{DBP}$	634	$\mathcal{AL}^{(\mathcal{D})}$	42	38	68	100
K_{STLab}	260	$\mathcal{ALCHIN}$	36	27	0	86
$K_{fisheries}$	963	$\mathcal{SI}$	9	5	0	70

Table 5.1: Input KBs: statistics

5.1 DBpedia subsets

5.1.1 DBpedia

DBpedia is a large and cross-domain knowledge base, which results from an effort to extract structured data out of Wikipedia. It is also often viewed as the core of the Linked Open Data Cloud, providing *de facto* reference IRIs, which are reused in numerous other datasets.

The 2014 release of the English version of DBpedia counts more than 4.5 million individuals, most of which correspond to a Wikipedia entry. For instance, the DBpedia individual denoted by the IRI `http://DBpedia.org/resource/Woody_Allen` is the one described by the Wikipedia page `en.wikipedia.org/wiki/Woody_Allen`. A large part of the ABox of DBpedia is automatically extracted from Wikipedia infoboxes, resulting in approximately 88 million ABox statements, which involve these individuals as well as approximately 800 atomic concepts and 3000 object or datatypeProperties. A (partial) mapping from the types used in Wikipedia infoboxes to the atomic concepts of DBpedia allows for the generation of formulas of the form $A(e)$, with A a DL atomic concept and e an individual. This mapping can be freely

updated by users. Similarly, an editable mapping from the properties used in infoboxes to DBpedia objectProperties allows for the generation of formulas of the form $R(e_1, e_2)$, with R a DL role, and e_1 and e_2 two individuals. A third editable mapping from infobox properties to DBpedia datatypeProperties allows for the generation of datatypeProperty assertions. Among the 88 million ABox statements, approximately 28 million are issued from the mapping to atomic concepts, and approximately 60 million from the mappings to objectProperties and datatypeProperties.

In addition to the ABox, a lightweight TBox has been manually crafted and updated, whose signature consists of atomic concepts, datatypeProperties and objectProperties appearing in the ABox. It can also be partly edited by users, adding subsumption axioms between two atomic concepts A and B (i.e. DL statements of the form $A \sqsubseteq B$), selecting an atomic concept A as the domain of an objectProperty R ($\exists R.\top \sqsubseteq A$), or as its range ($\top \sqsubseteq \forall R.A$), and selecting an atomic concept as the domain of a datatypeProperty, or a datatype (e.g. integers) as its range.

The 2014 version of this TBox counts 6730 logical axioms, 5647 of which are of the types just described, the least expressive corresponding DL being $\mathcal{ALCHF}^{(\mathcal{D})}$. An important number of atomic concepts are only subsumed by `owl:Thing` (i.e. $\top$ in DLs), including relatively specific ones, like "Festival", "Polyhedron" or "Fuel Type", which is probably inherent to the participative nature of this TBox.

Among the remaining axioms, 932 and 32 respectively form the objectProperties and datatypeProperties hierarchies, which cannot be freely edited. Domain and range axioms are generally expressed for the most specific properties in these two hierarchies, the more abstract properties being left without a domain or range.

67 OWL equivalentClass axioms ($A \equiv B$), and 3 OWL equivalentObjectProperty axioms ($R \equiv S$) respectively map DBpedia atomic concept and DBpedia objectProperties to atomic concepts and objectProperties of the lightweight ontology

Schema.org.[1]

The 49 remaining axioms are composed of 29 functional restriction for datatypeProperties (expressing the fact that a datatypeProperty cannot have more that one value for a given individual), and 20 disjointness axioms between atomic concepts ($A \sqsubseteq \neg B$). From a logical point of view, only these 49 axioms express some form of negation. The 29 cardinality restrictions on datatypeProperties fall outside the scope of this work, as explained in Section 5.1.2. The use of disjointness axioms on the other hand is very incomplete, offering a good illustration of their overall sparseness within the Linked Open Data Cloud. Some of these 20 disjointness axioms hold between some of the more abstract concepts of the TBox, which seems like a rational design choice (e.g. $\texttt{TimePeriod} \sqsubseteq \neg\texttt{Person}$, $\texttt{Event} \sqsubseteq \neg\texttt{Person}$, $\texttt{Place} \sqsubseteq \neg\texttt{Agent}$). But others hold between seemingly randomly chosen concepts (e.g. $\texttt{Tower} \sqsubseteq \neg\texttt{Person}$, $\texttt{UnitOfWork} \sqsubseteq \neg\texttt{Person}$, $\texttt{GeologicalPeriod} \sqsubseteq \neg\texttt{Person}$, $\texttt{Dog} \sqsubseteq \neg\texttt{Fish}$). Furthermore, 14 of these 20 axioms imply the concept `Person`.

Unsurprisingly, although important and successful efforts have been devoted to improve the quality of DBpedia over the years, it is still very easy to find DBpedia statements (within the ABox alone, or within the ABox and the TBox) which may make sense individually but, considered together, violate elementary common sense, as illustrated by example 1.1.1 in Chapter 1. The size of the whole dataset can be given as a first explanation, making it difficult to ensure a unique interpretation of each individual, atomic concept, datatypeProperty or objectProperty. The fact that almost all the content of DBpedia can be freely edited, either directly (TBox) or indirectly (Wikipedia infoboxes) probably plays an important role too. Finally, because negation is not used in the ABox, and remains very sparse in the TBox, it is also very frequent that these seemingly absurd sets of statements are logically

[1] http://schema.org/

consistent and coherent.

5.1.2 DatatypeProperties

Modeling errors within DBpedia can be due to incorrect values of datatypeProperty assertions. For instance, the value of a datatypeProperty assertion may be an integer whereas a floating point number is expected instead, or this value may be a very implausible one (like a 20 digits integer for the population of a country). But as explained in Chapter 2 Section 2.3.1, the detection and repair of such cases fall out of the scope of this work. This is why the values of datatypeProperty assertions are not taken into consideration in the following chapters. Instead, datatypeProperties are treated as atomic DL concepts, such that for instance the OWL statement `populationTotal`(*Bahia*, 14175341(xsd:integer)) is treated from a logical point of view as the DL ABox formula $\texttt{Has}_{\texttt{populationTotal}}(Bahia)$, where $\texttt{Has}_{\texttt{populationTotal}}$ is an atomic concept standing intuitively for "has a total population", regardless of the value.

5.1.3 Extraction procedure

Taking the whole DBpedia as input to the debugging strategies presented in the following sections is clearly prohibitive. In particular, they rely on (standard or less standard) reasoning algorithms whose execution is often intractable in the worst case, as explained in Chapter 1 Section 2.3.8.

Instead, the prototypical application scenario is a relatively small KB (up to a few thousand axioms), built for a specific application by aggregating data available on the SW, for instance out of a given signature. This is why subsets of DBpedia were automatically extracted and used as inputs of the upcoming experiments. Another

application scenario is the one where a TBox only needs to be retrieved for a given application, (and will prototypically be "populated" with local data, e.g. with a serialized database). In this scenario too, some ABox statements need to be retrieved, but only for debugging purposes. They will not be integrated to the KB eventually, but provide evidence in order to identify common sense violations in the TBox.

An additional precaution was taken in order to extract plausible DBpedia subsets for a given application, which pertains to topical cohesion. It is indeed quite unlikely for a same application to require information about very different domains, like embryology and religion for instance. For this reason, the DBpedia subset extraction procedure was preceded by the automatic selection of a set E of "seed" individuals for a given topic. The algorithm used to compute E relies on so-called Wikipedia categories, and is detailed in Section 5.1.3.2.[2]. Then this set E of seed individuals is used as an input DL signature in order to extract DBpedia axioms in a more expected and syntactic fashion, described in Section 5.1.3.3.

5.1.3.1 Thematic cohesion

5.1.3.1.1 Wikipedia categories Wikipedia categories are labels assigned to Wikipedia pages (and therefore also to DBpedia individuals), and organized in a hierarchy, comparable to a thesaurus. These categories are not concepts in the DL sense, but should rather be viewed as keywords. In particular, the supercategory relation is not meant to be a subsumption relation. For instance, the Wikipedia category "Biologists" has "History of Biology" as a supercategory, but this is not supposed to mean that biologists are histories of biology. Such categories are natural candidate inputs when trying to identify topically related Wikipedia pages, and thus

[2]Wikipedia categories are not integrated to the extracted DBpedia subset, but only used temporarily to compute E, as explained in Section 5.1.3.1.1.

DBpedia individuals, as opposed to the atomic concepts of the DBpedia TBox, which tend to cross multiple domains. This is why Wikipedia categories were used in order to select a topically coherent set E or seed DBpedia individuals, which was in turn used as an input signature for a more traditional axiom extraction procedure. But because Wikipedia categories are not DL concepts, they were not integrated to the extracted DL KBs.

5.1.3.2 Seed individuals selection

This section details the algorithm used to select a topically coherent set E of seed DBpedia individuals. It takes as input a Wikipedia category w_0, for instance "Tourism" or "History of Biology".

Two practical issues are addressed by this algorithm. The first issue is that thematic cohesion may sometimes be lost when navigating through the Wikipedia supercategory relation. For instance, both "Modeling" and "Public Service Announcements" are immediate subcategories of "Advertising", such that their respective immediate subcategories "Murdered Models" and "Copyright Announcements" are closely related in the hierarchy, although they have arguably nothing in common. The second issue is in a sense the converse of the first one: two Wikipedia categories may be closely related topically, but not in the hierarchy.

To address these two issues, a specific algorithm was designed, which consists of a series of successive expansion and filtering phases of two (disjoint) sets W and E of Wikipedia categories and individuals respectively. W is initiated with all subcategories of w_0 in the Wikipedia category hierarchy up to depth m_1, with m_1 a fixed parameter, and E with all DBpedia individuals tagged with some $w \in W$.

In order to address the first issue, i.e. the lack of topical cohesion, W and E

are both filtered as follows. Within E, only individuals tagged with at least m_2 categories of W are retained (with m_2 another parameter). Then only the categories tagging one of the remaining individual are retained in W (or in other words, a Wikipedia category is retained iff it shares a tagged individual with at least $m_2 - 1$ other categories in W), based on the intuition that core categories for the topic under consideration should share tagged individuals.

In order to address the second issue, i.e. missing categories/individuals for the topic under consideration, W is then expanded with all other Wikipedia categories tagging some $e \in E$. And E with all individual tagged by some $w \in W$.

A second filtering phase, identical to the first one, is then applied to W and E.

This yields two sets W and E which have been successively filtered, expanded, and filtered again.

5.1.3.3 Axiom extraction procedure

Given a set E of seed DBpedia individuals for the topic under consideration, the axiom extraction procedure starts with all DBpedia ABox statements involving some $e \in E$. Then for each individual e' appearing in one of these axioms, all type declarations for e' (i.e. formulas of the for $A(e')$, with A an atomic concept) are retrieved. Finally, for all datatypeProperties, objectProperties and atomic concepts appearing in these statements, the corresponding domains/ranges and all transitively subsuming atomic concepts in the DBpedia TBox are retrieved as well.

An additional precaution is taken for the ABox. The dataset available from the DBpedia endpoint (`http://DBpedia.org/sparql`) has been procedurally closed under the rule $\{A(e), A \sqsubseteq B\} \vdash_d B(e)$, with A and B atomic concepts, e an individual, and $A(e)$ and $A \sqsubseteq B$ two axioms. In other words, all consequences of DBpedia

which could be derived by application of this rule have been added to it as axioms. But the focus here is on the identification of erroneous axioms understood as explicitly *asserted* statements, and not automatically derived ones. Therefore, if K is the extracted KB, each axiom ϕ such that $K \setminus \{\phi\} \vdash_d \phi$ holds is removed from K.

5.1.4 List of DBpedia subsets

This section describes the five DBPedia subsets. The main statistics about these KBs are also reproduced in table 5 above, and the KBs are available online.[3]

- K_1^{DBP} is the output of the above seed individuals selection and axiom extraction procedures, with parameters $m_1 = m_2 = 2$, and starting with the Wikipedia category "Tourism" as w_0.

  K_1^{DBP} is composed of 8329 logical axioms, the least expressive underlying DL being $\mathcal{AL}^{(\mathcal{D})}$. Its signature counts 106 atomic OWL classes (i.e. DL atomic concepts), 100 objectProperties (i.e. DL roles), 205 datatypeProperties (considered as DL atomic concepts here, as explained above in Section 5.1.2) and 1437 individuals.

- K_2^{DBP} is the output of the same extraction procedure, with parameters $m_1 = m_2 = 2$ as well, but starting with the Wikipedia category "Advertising" as w_0.

  K_2^{DBP} is composed of 6148 logical axioms, the least expressive underlying DL being $\mathcal{AL}^{(\mathcal{D})}$. Its signature counts 77 atomic OWL classes, 69 objectProperties, 116 datatypeProperties and 866 individuals.

- $K_{1.1}^{DBP}$ is a small subset of K_1^{DBP}. It was created as a benchmark, in order

[3] http://juliencorman.github.io/

to obtain an evaluation of recall with DBpedia axioms as input.[4] The small size of $K_{1.1}^{DBP}$ allowed for an extensive manual review of its axiom, such that a debugging procedure could be evaluated based on its ability to spot exactly the axioms evaluated as erroneous. This is hardly feasible K_1^{DBP} and K_2^{DBP}, which count several thousands of axioms each.

$K_{1.1}^{DBP}$ was automatically extracted from K_1^{DBP} (and not from DBpedia directly), by applying the axiom extraction procedure described in Section 5.1.3.3, but with a single input seed individual, namely *Smithsonian Institution*. This individual was selected due to its frequency within K_1^{DBP}.[5]

$K_{1.1}^{DBP}$ is composed of 225 logical axioms, the least expressive underlying DL being $\mathcal{AL}^{(\mathcal{D})}$. Its signature counts 27 atomic OWL classes, 17 objectProperties, 38 datatypeProperties and 29 individuals.

- $K_{1.2}^{DBP}$ was extracted from K_1^{DBP} by applying the same procedure as for $K_{1.1}^{DBP}$, but with a different seed individual, namely *Mohammed-El-Fayed*, which was selected for the same reason as previously.

  $K_{1.2}^{DBP}$ is composed of 159 logical axioms, the least expressive underlying DL being $\mathcal{AL}^{(\mathcal{D})}$. Its signature counts 16 atomic OWL classes, 11 objectProperties, 18 datatypeProperties and 28 individuals.

- $K_{1.3}^{DBP}$ was obtained by applying the same procedure as for $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$, but with 7 seed individuals, which were the 7 most frequent ones in K_1^{DBP} (this includes the two seed individuals for $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$ respectively).

[4] The automatically degraded datasets presented in Section 5.2 also allow for an evaluation of recall, but are more artificial.

[5] More exactly, it was selected due to the fact that it appears in a large number of axioms, *and with different roles* if these axioms are of the form $R(e_1, e_2)$. The exact ordering of individuals of K_1^{DBP} based on this adjusted frequency is described in Chapter 7 Section 7.3.2.

$K_{1.3}^{DBP}$ is composed of 634 logical axioms, the least expressive underlying DL being $\mathcal{AL}^{(\mathcal{D})}$. Its signature counts 42 atomic OWL classes, 38 objectProperties, 68 datatypeProperties and 100 individuals.

5.2 Automatically degraded KBs

This section presents the KBs which were automatically degraded for some of the following experiments, as well as the degrading procedure. In order to prevent possible misunderstandings, it should be emphasized that automatically degraded KBs are not realistic input bases. They were simply created in order to obtain an arguably artificial but objective evaluation protocol.

The input KBs (before being degraded) were selected for their high quality, and extended with random axioms built out of their respective signatures. The two underlying assumptions are on the one hand that the input KB K before being degraded does not contain obvious violations of common sense, and that the randomly generated axioms Φ on the other hand are very likely to be intuitively absurd wrt to K. A debugging strategy is then evaluated based on its ability to automatically spot the axioms of Φ within $K \cup \Phi$.

A third related assumption is made wrt the consequences of the extended KB $K \cup \Phi$. If K and $K \cup \Phi$ are both consistent, it is assumed that most newly generated consequences, i.e. $\mathrm{Cn}(K \cup \Phi) \setminus \mathrm{Cn}(K)$, are very likely to be absurd wrt K as well.

Section 5.2.1 introduces the KBs before they were degraded, and Section 5.2.4 presents the degrading procedure.

5.2.1 Input KBs

This section describes the two input KBs before they were automatically degraded. They were both selected for their high quality, and because their signatures contain individuals with linguistic labels.

As just explained, for the evaluation procedure to be effective, the input KBs before being degraded should not contain common sense violation of the type described in Chapter 1, which rules out here the use of (fragments of) large and/or partially crowdsourced KBs such as DBpedia or Freebase.

5.2.2 STLab gold standard

The first input KB will be called K_{STLab}. It is built out of a gold standard described in [GNP+12], which was itself created in order to evaluate the automatic entity typing tool presented in the same article. This gold standard is composed of 100 manually typed Wikipedia entities (i.e. Wikipedia pages). It has been built collaboratively with a web-based application allowing annotators to discuss their decisions. It is based on the top-level ontology DOLCE+DnS Ultralite (abbreviated as DUL), available at `http://www.ontologydesignpatterns.org/ont/dul/DUL.owl`. DUL is itself a simplification and an extension of the top-level ontology DOLCE (described in [MBG+03]).

For each Wikipedia entity r of the gold standard, the annotators had to determine whether r was a concept or an individual, and also had to map r to an atomic concept A of DUL.[6] If r was categorized as an individual, then the mapping corresponds to a statement of the form $A(r)$, where A is the most specific concept of DUL with

[6] A mapping to Wordnet synsets was also performed, but not used for the experiments described in this thesis.

r as an instance. If r was categorized as a concept, then the mapping corresponds to a statement of the form $r \sqsubseteq A$ where A is the most specific concept of DUL subsuming r. All typing decisions were taken with an inter-annotator agreement superior to 70 %. The gold standard is available at http://stlab.istc.cnr.it/stlab/WikipediaOntology/.

Only the entities considered as individuals are retained for the experiments described in the following chapters, i.e. 86 out of the 100 entities of the gold standard, such that the ABox of the dataset K_{STLab} is composed of 86 statements of the form $A(r)$, where r is an individual and A an atomic concepts from DUL. The TBox of K_{STLab} is a locality-based module [CGHKS08] extracted from DUL out of these atomic concepts. The linguistic labels of the 86 individuals were retrieved from DBpedia, using the mapping from Wikipedia pages to DBpedia entities.

In total (ABox + TBox), K_{STLab} counts 260 logical axioms, the least expressive underlying DL being $\mathcal{ALCHIN}$. Its signature is composed of 36 atomic OWL classes, 27 objectProperties, no datatypeProperty and 86 individuals.

5.2.3 Fragment of the NeOn fisheries ontology

The second input KB will be called $K_{fisheries}$, and is a fragment of the fisheries ontology network used in the NeOn project.

NeOn[7] was a 2006-2010 European research project focused on the use of ontologies for large-scale semantic applications, with an emphasis on Semantic Web standards and technologies. The fisheries case study, available online,[8] is one of the two "ontology networks" used as "the backbones of the applications" developed within the NeOn project. The network consists of 9 OWL DL KBs with overlapping signatures.

[7] http://www.neon-project.org
[8] http://www.neon-project.org/nw/Ontologies

$K_{fisheries}$ is a small subset of it, automatically extracted with the procedure described in Section 5.1.3.3. starting with 10 randomly selected seed individuals, and ignoring datatypeProperties.

$K_{fisheries}$ is composed of 963 logical axioms, and the least expressive underlying DL is $\mathcal{SI}$. Its signature is composed of 9 atomic OWL classes, 5 objectProperties, no datatypeProperty and 70 individuals (mostly geographical or administrative entities).

5.2.4 Degrading procedure

Let K_0 designate the (consistent) input KB before being degraded. The degrading procedure proceeds by extending K_0 n times incrementally with a new randomly generated axiom. $K_1, .., K_n$ will designate the successive extensions of K_0, and $\phi_1, .., \phi_n$ the successively generated axioms.

A random axiom ϕ_i is simply generated by randomly selecting a source axiom $\phi \in K_i$, and replacing each element of the signature of ϕ by a random element of $\text{sig}(K_i) = \text{sig}(K_0)$, preserving the syntactic structure of ϕ. So each elements of $N_{Ind}(\phi)$, $N_{Con}(\phi)$ or $N_{Role}(\phi)$ is respectively replaced by a random element of $N_{Ind}(K)$, $N_{Con}(K)$ or $N_{Role}(K)$. For instance, if $\phi = A \sqsubseteq \forall R.B$, with $A, B \in N_{Con}(K)$ and $R \in N_{Role}(K)$, then $\phi_i = A' \sqsubseteq \forall R'.B'$, with $A', B' \in N_{Con}(K)$ and $R' \in N_{Role}(K)$ as well.

Two additional requirements are put on the axiom generation procedure. First, the extended KB must be consistent, i.e. $K_{i-1} \cup \{\phi_i\} \nvdash \bot$.

The second requirement pertains to the consequences of the extended KB K_i. As explained in Chapter 4 Section 4.2.3, the debugging procedures which incorporate linguistic evidence rely on so-called *plausibility scores* computed for some consequences

of each candidate output base. The eligible consequences, i.e. the ones for which a score may be computed, are defined syntactically (up to equivalence): it may be all consequences of the form $A(e)$ or $\neg A(e)$, with A an atomic concept and e an individual, or the former plus all consequences of the form $\exists R.\top(e)$, $\exists R^{-}.\top(e)$, $\forall R^{-}.\bot(e)$ and $\forall R.\bot(e)$, wih R an atomic role, as explained in Chapter 4 Section 4.2.3. Therefore it is required for each K_i that at least one new consequence of this syntactic form is generated, i.e. if Ψ is the set of formulas of this form which can be built out of the signature of K, then it is required that $(\mathrm{Cn}(K_{i-1} \cup \{\phi_i\}) \setminus \mathrm{Cn}(K_{i-1})) \cap \Psi \neq \emptyset$.

Note that if the initial input base K_0 is successively extended with strictly more than one axiom (i.e if $n > 1$), this second requirement still does not guarantee that all subbases of K_n can be distinguished wrt to linguistic evidence, because there may be two subbases Q_1 and Q_2 of K_n such that $\mathrm{Cn}(Q_1) \cap \Psi = \mathrm{Cn}(Q_2) \cap \Psi$.

This second requirement does not guarantee either that a plausibility score will actually be computed for each new consequence of this form. In particular, it may be the case for a new consequence $C(e) \in (\mathrm{Cn}(K_n) \setminus \mathrm{Cn}(K_0)) \cap \Psi$ that no linguistic occurrence of the label of e could be retrieved, or that the support set (see Chapter 4 Section 4.2.3) for $C(e)$ is empty.

The number of automatically degraded KBs varies from one experiment to another, as well as the number of randomly generated axioms for each degraded KB, therefore they will be given with the descriptions of each experiment.

5.3 Linguistic corpora

This section explains how linguistic corpora were automatically retrieved for each of the seven input KBs described above.

5.3.1 Homonymy

As explained in Chapter 4, the linguistic evidence gathered to debug an input KB K consists of distributional representations of the labels of the individuals of $\text{sig}(K)$. Among other reasons (listed in Chapter 4 Section 4.2.1), this choice is motivated by the fact that individual linguistic labels are prototypically proper names, like "Woody Allen" or "Egypt", and as such are generally less ambiguous than the linguistic labels of atomic concepts, which may be as vague as "function", "group" or "model". But this obviously does not completely solve the polysemy issue. In particular, proper names may have homonyms. For instance, "JFK" may designate a politician or an airport.

Therefore, in order to avoid possible noise during the debugging process, if an individual $e \in N_{Ind}(K)$ is labeled with at least one (English) label l considered as potentially polysemous, then none of the labels of e is used as a source of linguistic evidence during the debugging process. The axioms involving e are not removed from K though, which would bias the evaluation. In other words, individuals with at least one potentially polysemous label are treated like individuals without label.

In order to determine which labels are potentially polysemous, three simple but relatively efficient heuristics are applied, which are also applicable to a large number of other potential input SW KBs, and therefore can be considered as an integral part of the debugging process. First, a linguistic label l is considered potentially polysemous if more than 10 million pages[9] can be retrieved by a web search engine using l (quoted) as a query. The second heuristic consists in ruling out l if two DBpedia individuals e_1 and e_2 are labeled with it, and if they are not considered as identical ($e_1 = e_2$ in DLs) within DBpedia. The third heuristic is based on the IRI of the

[9]This threshold of 10 million was chosen on a empirical basis.

DBpedia individual(s) whose label(s) contain l as a substring: if such an IRI contains parentheses, then l is considered as potentially polysemous. For instance, let l be the label "Mekong Delta". There is at least one DBpedia individual whose linguistic label contains l as a substring, and whose IRI contains parentheses, namely the individual `http://DBpedia.org/resource/Mekong_Delta_(band)`, which is likely to indicate that the label "Mekong Delta" is polysemous.

Although the two last heuristics rely on DBpedia as an external resource, they can be applied to any input KB with linguistic labels for individuals, i.e. the input KB doesn't need to be a subset of DBpedia, neither to share part of DBpedia's signature.

5.3.2 Corpus retrieval

For each input KB K, for each individual $e \in N_{Ind}(K)$, if e had an English label, and if no label of e was considered potentially polysemous, then 200 or 400 web pages were retrieved for e with a search engine, using the (quoted) labels of e as queries (200 pages per individual for the larger KBs K_1^{DBP} and K_2^{DBP}, and 400 pages per individual for the other KBs). If e had several labels, the 200 (or 400) pages were distributed over these labels.

A few additional filters were added in order to rule out some urls (for instance `http://www.ebay.com`), as well as non-English texts. The set of retrieved pages were then cleaned thanks to the Bootcat library [BB04], allowing for the removal of most non-textual content, but also of many duplicated texts.

Two versions of each corpus were produced, as described in Chapter 4 Section 4.3.3: one of them is just tokenized, whereas the other one is analyzed linguistically with the Stanford Tagger [TKMS03]. As already explained, the motivation behind

the non linguistically processed version is the possibility to use a language model as an external resource, in order to estimate the prior probability of a linguistic context c observed in the corpus, represented as an n-gram. The higher this probability, the less meaningful is the observation that this context is shared by two individuals. The Microsoft Web N-gram Services[10] were used for this purpose. In particular, they provide smoothed n-gram probabilities computed out of very large collections of indexed web pages. This was particularly relevant for the problem at hand due of the nature of the corpora, because many of the retrieved web pages contain procedurally generated web content. Many of the linguistic contexts shared by several individuals of the different input KBs turn out to be procedurally generated, for instance "learn more about $< X >$" or "$< X >$ on Twitter". Most of these contexts are not meaningful as far as selectional preferences are concerned, and n-gram probabilities offer a relatively efficient way to make such contexts less informative when computing distributional representations of the individuals of the input KB.

5.4 Conclusion

This chapter introduced the datasets (KBs and linguistic corpora) used for the different evaluations presented in the following chapters. The choice was made to present all datasets in a single chapter, because some of them are used in multiple evaluations.

Five of the KBs used for these evaluations are automatically extracted subsets of DBpedia (ABox and TBox), of different sizes (159 to 8329 axioms), which, because of the size and collaborative nature of DBpedia as a whole, are likely to contain violation of common sense. An introduction to DBpedia was provided in Section

[10] http://research.microsoft.com/en-us/collaboration/focus/cs/web-ngram.aspx

5.1, followed by a description of the procedure applied to extract these five KBs, which was conceived as an attempt to favor topical cohesion.

A more artificial but more objective evaluation protocol was also designed as an alternative, based on the automated degradation of a KB with axioms randomly generated out of its signature. The axiom generation procedure was described in Section 5.2, as well as the two KBs used for this purpose, of an arguably higher quality (before being degraded) than subsets of DBpedia.

Finally, Section 5.3 described the procedure applied to extract linguistic corpora for each of these KBs from automatically retrieved web pages, and in particular the way individuals with potential homonyms were handled.

Chapter 6

Trimming a consistent KB

This chapter investigates the use of linguistic evidence in order to identify implausible consequences of a consistent input KB K, and to suggest axioms to be discarded (or at least amended) accordingly, or equivalently to select the complement in K of these discarded axioms as an output base.

Section 6.1 evaluates the linguistic plausibility score defined in Chapter 4 Section 4.2.3, and Section 6.2 evaluates the identification of one faulty axiom within K, based on the four preference relations over the family $\mathcal{Q}$ of candidate output bases defined in Chapter 4 Section 4.2.5.

Then Section 6.3 discusses the identification of multiple faulty axioms within K. It shows that computing all optimal elements of $\mathcal{Q}$ according to any of these four preference relations may be costly if $\mathcal{Q} = 2^K$. Section 6.3.2.1 proposes instead a simple greedy approach, evaluated with real and automatically degraded data. Section 6.3.2.2 addresses the exact computation of all optimal elements of $\mathcal{Q}$ wrt one of the four abovementioned preference relations, namely $\preceq_{lex_K}$. An algorithm is provided, whose correctness is proven, together with worst-case complexity bounds

for the problem, showing that, if Ψ_K is the set of consequences of K with a plausibility score, then for a fixed $|\Psi_K|$, the problem is as hard as computing all maximal subsets of K not entailing ψ for each $\psi \in \Psi_K$.

For readability, all proofs of propositions made in the current chapter are grouped in Section 6.5.

6.1 Plausibility: experiments

This section presents an evaluation of the plausibility score defined in Chapter 4 Section 4.2.3. As a reminder, given a set Q of axioms, as well as a set $\Psi_Q \subseteq \mathrm{Cn}(Q)$ of consequences of Q to be evaluated, for each $\psi \in \Psi_Q$, the plausibility score $\mathrm{sc}_Q(\psi)$ estimates to what extent ψ is likely to hold if the rest of Ψ_Q does, or equivalently whether ψ can be viewed as an outlier within Ψ_Q.

In order to avoid possible confusions, plausibility scores do not offer by themselves an effective way to debug a KB. In particular, they estimate the plausibility of consequences, not axioms. But all debugging strategies guided by linguistic evidence proposed in this thesis rely on these scores, at a lower level. Therefore it is interesting to evaluate plausibility independently, in order to see if it provides an accurate basis for actual debugging techniques, such as the identification of a faulty axiom, evaluated in Section 6.2, or more elaborated algorithms, like the ones presented in Section 6.3 and Chapter 8. In other words, the current section should be viewed as an isolated evaluation of a lower-level component, and not as an evaluation of a complete debugging strategy.

In order to evaluate whether these linguistic plausibility scores are accurate or not, automatically degraded KB were generated out of the datasets $K_{\mathit{fisheries}}$ and K_{NeOn}, according to the procedure described in Chapter 5, Section 5.2.4. For each

$K \in \{K_{STLab}, K_{fisheries}\}$, 100 random axioms $\phi_1, \ldots, \phi_{100}$ were generated out of the signature of K, and each of them added independently to K, yielding 100 degraded KBs $K_1, \ldots, K_{100}$, such that for each $1 \leq i \leq 100$, $|K_i| = |K| + 1$.

Each input KB K before being degraded was selected for its quality, and the assumption was made that it did not not contain obvious violations of common sense. On the other hand, randomly generated axioms are very likely to be absurd wrt K. Therefore one may also expect proper consequences of a degraded KB K_i to convey absurd information (for instance $\texttt{Architect}(\mathit{Belgium})$), or at least to be outliers within $\mathrm{Cn}(K_i)$ (like $\texttt{Person}(\mathit{CEO})$ in example 1.1.1 Chapter 1). More specifically, linguistic plausibility was evaluated based on its ability to identify the formulas of $\Psi_{K_i} \setminus \Psi_K$ as the least plausible ones within Ψ_{K_i}.

The choice was made to generate only one axiom for each K_i in order to limit the cardinality of $\Psi_{K_i} \setminus \Psi_K$.[1] Otherwise, it may be the case that consequences of Ψ_K become the outliers within Ψ_{K_i}. Actually, as will be discussed in more details, the few degraded KBs for which the plausibility score performed poorly in the following experiments fall into this category.

6.1.1 Distributional settings

Distributional representations were built for all named individuals in $\mathrm{sig}(K)$, out of the automatically retrieved corpora described in Chapter 5 Section 5.3. As explained in Chapter 4 Section 4.3.3, two different forms of linguistic contexts were alternatively tested, either n-grams preceding or following an occurrence of an individual's label, or sequences of lemmatized words surrounding an occurrence. All frequency weighting functions listed in Chapter 4 Section 4.3.4 were also evaluated. The pos-

[1] Obviously, this does not guarantee that the cardinality of $\Psi_{K_i} \setminus \Psi_K$ will be low, but it increases the probability that it will.

sible combinations of contexts and weighting functions will be designated with the following acronyms: LP for lemmatized contexts weighted with PPMI, and NP, NS and NPS for n-grams weighted with PPMI, self-information and both respectively.[2]

6.1.2 Results

For each K_i and each $\psi \in \Psi_{K_i}$, the plausibility $\mathrm{sc}_{K_i}(\psi)$ was obtained as defined in Chapter 4 Section 4.2.3, and then Ψ_{K_i} was ordered by increasing plausibility,[3] such that the formula ranked as number 1 has the lowest plausibility score. As a notational shortcut, consequences issued from random axiom generation within Ψ_{K_i} will be denoted with Ψ_{rand_i}, i.e. $\Psi_{rand_i} = \Psi_{K_i} \setminus \Psi_K$. The evaluation is successfull iff these formulas have low plausibility scores among all formulas of Ψ_{K_i}. For readability, Ψ_{rand_i} will occasionally be paraphrased as the set of "random consequences" in Ψ_{K_i}, although this is arguably an improper denomination (axioms were randomly generated, whereas formulas of Ψ_{rand_i} are just consequences of the extended KB K_i which are not consequences of K).

If Q is a set of axioms, then Ψ_Q in the rest of this section will designates all consequences of the form $A(e)$ or $\neg A(e)$ for which a plausibility score can be computed, with A an atomic DL concept and e an individual.

Table 6.1 provides statistics about $\Psi_{K_{fisheries}}$ and $\Psi_{K_{STLab}}$. Row $\mu(|\Psi_{K_i}|)$ gives the cardinality of Ψ_{K_i} on average for all 100 degraded KBs. For instance, the value 216.1 for the KB $K_{fisheries}$ means that there were on average 216.1 formulas in each Ψ_{K_i}, i.e. 216.1 consequences of the form $A(e)$ or $\neg A(e)$ were evaluated on average in each of the 100 degraded versions K_1, .., K_{100} of $K_{fisheries}$. Then row $\mu(|\Psi_{rand_i}|)$ gives the

[2] As a reminder (see Chapter 4 Section 4.3.3), self-information could not be used as a weighting function for lemmatized contexts, because language models are based on n-grams.

[3] The ranking was a strict ordering: if two consequences had the same score, one of them was randomly designated as strictly lower ranked.

	$K_{fisheries}$	K_{STLab}
$\mu(\|\Psi_{K_i}\|)$	216.1	620.27
$\mu(\|\Psi_{rand_i}\|)$	2.93	20.71
$-/+\sigma(\|\Psi_{rand_i}\|)$	11	6

Table 6.1: Evaluated consequences for all 100 degraded versions of K_{STLab} and $K_{fisheries}$

average number of consequences within each Ψ_{K_i} which were issued from random axiom generation, i.e. the average value of $|\Psi_{K_i} \setminus \Psi_K|$. For instance, the value 2.93 for the KB $K_{fisheries}$, means that there were on average 2.93 "random consequences" among the 216.1 evaluated ones for each of the 100 degraded versions K_1, .., K_{100} of $K_{fisheries}$. The goal of the evaluation was to identify these "random consequences" as the least plausible ones.

These values are slightly misleading though, because they do not account for the variation of $|\Psi_{rand_i}|$ over the 100 random degraded KBs. Let $\sigma(|\Psi_{rand_i}|)$ designate the standard deviation of the observed values of $|\Psi_{rand_i}|$ for each $i \in \{1, 2, .., 100\}$. Then row $-/+\sigma(|\Psi_{rand_i}|)$ gives the number of values of i (i.e. the number of degraded KBs) for which $|\Psi_{rand_i}|$ was either inferior to $\mu(|\Psi_{rand_i}|) - \sigma(|\Psi_{rand_i}|)$, or superior to $\mu(|\Psi_{rand_i}|) + \sigma(|\Psi_{rand_i}|)$. This gives an indication of the distribution of $|\Psi_{rand_i}|$ for $i \in \{1, 2, .., 100\}$. The values are small (in comparison, for a normal distribution, this value would be 31.8), illustrating the fact that for most degraded KBs, $|\Psi_{rand_i}|$ was very low. For instance, for $K_{fisheries}$, the average value of 2.93 does not account for the fact that for most degraded KBs (75/100), there was only one "random consequence".

Table 6.2 provides the results of the evaluation of plausibility based on Ψ_{K_i}. Column "bRank" gives the average ranking within Ψ_{K_i} of the formula of Ψ_{rand_i} with lowest plausibility, and column "wRank" the average ranking of the formula of Ψ_{rand_i} with highest plausibility. The lower these rankings, the more efficient the plausibility

	$K_{fisheries}$			K_{STLab}		
	brank	wrank	p-val	brank	wrank	p-val
LP	4.15/216.1	14.02/216.1	<0.001	63.75/620.27	360.4/620.27	<0.001
NP	9.73/216.1	16.11/216.1	<0.001	63.09/620.27	380.08/620.27	<0.001
NS	7.33/216.1	15.21/216.1	<0.001	68.42/620.27	360.55/620.27	<0.001
NPS	5.59/216.1	13.24/216.1	<0.001	61.58/620.27	380.37/620.27	<0.001

Table 6.2: Average ranking among Ψ_{K_i} of the lowest-ranked and highest-ranked formulas of Ψ_{rand_i}, and p-value for the rankings of all formulas of all Ψ_{rand_i}

score is at detecting consequences caused by random axioms. For instance, on average over the 100 degraded versions of $K_{fisheries}$, the ranking of the consequence of Ψ_{rand_i} with lowest plausibility was 4.15 over 216.1, and 14.2 over 216.1 for the consequence of Ψ_{rand_i} with highest plausibility. Column "pVal" gives the probability (t-test) for the cumulated rankings of all formulas in all Ψ_{rand_i} to be as low as the observed ones, if all consequences in all Ψ_{K_i} had been randomly ordered.

For $K_{fisheries}$, average results are very convincing with a significant p-value for all four distributional settings. But a closer look at the data reveals that results are actually much better in most cases. As an illustration, for most degraded KBs (75/100), there is only one formula in Ψ_{rand_i}, i.e. only one "random consequence". For the best distributional setting (LP), in most of theses cases (57/75), this formula is also the one with lowest plausibility in Ψ_{K_i}, among all consequences of Ψ_{K_i}, i.e. the only randomly generated consequence is also the least plausible one according to linguistic evidence. This is very encouraging, especially considering the relatively small number of individuals (71) in sig($K_{fisheries}$), i.e. the fact that the support set to evaluate the plausibility of a consequence $\psi \in \Psi_{K_i}$ is limited. On the other hand, performances are generally much worse (sometimes close to a random baseline) when the cardinality of Ψ_{rand_i} is important ($> 0.15 * |\Psi_{K_i}|$), which can be explained by the fact that support sets for some concepts were significantly altered after the extension

of K with ϕ_i.

For K_{STLab}, the results are clearly less satisfying, although still significantly positive. Several possible explanation may be suggested for these comparatively worse results. First, although the number of individuals in both cases (86 and 70) is comparable, individuals of $K_{\mathit{fisheries}}$ are mostly geographic or administrative entities, and therefore they tend to form relatively homogeneous groups (countries, international organizations, etc...). Comparatively, the individuals of K_{STLab} are extremely diverse (human beings, theater plays, biological species, times intervals, or even *copyleft*) such that on such a small sample (86), support sets of a significant size for a given concept can rarely be found.

Another possible explanation is the fact that the proportion of TBox axioms in K_{STLab} is very high (184/260, i.e 0.71, which is arguably unrealistic), and much lower for $K_{\mathit{fisheries}}$ (30/963, i.e. 0.03). This led to the generation of a much higher number $|\Psi_{rand_i}|$ of "random consequences" for K_{STLab}, on average 20.71 against 2.93, and up to 206 for a same single degraded KB. This last observation provides a possible insight about the effectiveness of the approach. Plausibility in theory may detect consequences caused by erroneous TBox axioms, and actually did in some cases. But based for these 200 KBs, it seemed more efficient at detecting consequences caused by randomly generated ABox axioms than by randomly generated TBox axioms.

As for the distributional settings, $K_{\mathit{fisheries}}$, the two most beneficial (but unfortunately incompatible) factors were the use of lemmatized contexts on the one hand (LP), and the queries over the Web n-gram corpus on the other hand (NS and NPS), whereas for K_{STLab}, none of the four configurations clearly stands out, neither positively nor negatively.

6.2 Trimming a single axiom: experiments

This section presents an evaluation of the four preference relations over candidate output bases defined in Chapter 4 Section 4.2.5, which are based on linguistic plausibility. As a reminder, these are total preorders over the family $\mathcal{Q}$ of candidate output KBs. Intuitively, if $\preceq$ designates one of these four relations, then $Q_1 \preceq Q_2$ iff Q_2 is more likely to hold than Q_1 given the linguistic input.

The evaluation presented in this section is restricted to the case where at most one axiom is to be discarded within the input KB, the case of a set of (strictly more than one) axioms being addressed in Section 6.3. Equivalently, each immediate subbase of the input KB is a candidate output, as well as the input KB itself. For instance, in example 4.2.2 in Chapter 4, one may expect the subbase $Q_2 = K \setminus (2)$ to be optimal among K and its immediate subbases, such that (2) is the best candidate axiom for removal.

The dataset for this experiment is the same as in Section 6.1 for the KB $K_{\mathit{fisheries}}$. Let $K = K_{\mathit{fisheries}}$. 100 random axioms $\phi_1, \ldots, \phi_{100}$ were generated out of the signature of K, and each of them added independently to K, yielding 100 degraded KBs $K_1, \ldots, K_{100}$, which served as input for the evaluation.

If K_i is a (degraded) input KB, then each axiom of K_i is a candidate for removal, and the randomly generated axiom ϕ_i is one of them. Equivalently, each immediate subbase of K_i (i.e. of cardinality $|K_i| - 1$) as well as K_i itself is a candidate output base, and because $K_i = K \cup \{\phi_i\}$, one of the immediate subbases of K_i is K. For each K_i, the family of candidate subbases will be designated with $\mathcal{Q}_i = \{Q_{i,1}, \ldots, Q_{i,|K_i|}\} \cup \{K_i\}$. Then there must be a $1 \leq j \leq |K_i|$ such that $Q_{i,j} = K$.

Because it has been randomly generated, the assumption is made that the axiom

ϕ_i is very likely to be absurd within K_i. So in order to evaluate whether the ranking of $\mathcal{Q}_i$ defined by a given preference relation is accurate, one can simply verify whether K is among the best ranked elements of $\mathcal{Q}_i$.

For each K_i, the family $\mathcal{Q}_i$ of candidate subbases was generated, and the elements of $\mathcal{Q}_i$ were ranked according to the four preference relations over candidate subbases defined in Chapter 4 Section 4.2.5, namely $\preceq_{\text{comp}}$, $\preceq_{\text{comp}_{K_i}}$, $\preceq_{lex}$ and $\preceq_{lex_{K_i}}$.[4] Table 6.3 gives the ranking of K within $\mathcal{Q}_i$ on average for all 100 K_i, for the LP distributional setting only (lemmatized contexts weighted with PPMI), because it gave the best results in the previous experiment. The lower this value, the more accurate the corresponding preference relation is. For instance, on average for all 100 K_i, the ranking of K among candidate subbases according to $\preceq_{\text{comp}}$ was 7.86.

An additional precaution was taken though in order to avoid artificially good results. For many $Q \in \mathcal{Q}_i \setminus \{K_i\}$, it was the case that $\Psi_Q = \Psi_{K_i}$, i.e. consequences with a plausibility score were identical for Q and K_i. In other words, no consequence of K_i with linguistic support was lost when weakening K_i to produce Q, such that from the definition of plausibility (Chapter 4, Section 4.2.3), for all $\psi \in \Psi_Q = \Psi_{K_i}$, $\text{sc}_Q(\psi) = \text{sc}_{K_i}(\psi)$. Therefore Q and K_i could simply not be distinguished based on plausibility scores. So if $\preceq$ is any of the four preference relations defined in Chapter 4 Section 4.2.5, according to these definitions, $\preceq$ defines a large equivalence class $\mathcal{V}_i \subset \mathcal{Q}_i$, such that for all $V \in \mathcal{V}_i$, $V \prec K_i$. But on the other hand, the random axiom generation procedure (presented in Chapter 5 Section 5.2.4) guarantees that $\Psi_K \subset \Psi_{K_i}$, and therefore that $K \notin \mathcal{V}_i$. So if $K_i \prec K$, which is most often the case, then for all $V \in \mathcal{V}_i$, $V \prec K$ also holds, such that K has an artificially good ranking within $\mathcal{Q}_i$. In order to avoid this bias, the ranking of K is given within $\mathcal{Q}'_i = \mathcal{Q}_i \setminus \mathcal{V}_i$ instead, or in other words there is only one candidate KB $Q \in \mathcal{Q}'_i$ such that $\Psi_Q = \Psi_{K_i}$,

[4] Again, the ranking was randomly turned into a strict ordering (see footnote 3).

	rank	p-val
$\preceq_{\text{comp}}$	7.86 / 80.03	< 0.001
$\preceq_{\text{comp}_{K_i}}$	8.05 / 80.03	< 0.001
$\preceq_{lex}$	6.51 / 80.03	< 0.001
$\preceq_{lex_{K_i}}$	2.47 / 80.03	< 0.001

Table 6.3: Average ranking of $K_{fisheries}$ within each $\mathcal{Q}_i$, and p-value for such a low average ranking

namely K_i itself. For instance, as just mentioned, the average ranking of K among all candidate subbases according to $\preceq_{\text{comp}}$ is 7.86. So because $|K| = |K_{fisheries}| = 963$, and because each $K_i = K \cup \{\phi_i\}$, $|K_i| = 964$, such that $|\mathcal{Q}_i| = 965$, i.e. there are 965 candidate output bases for each K_i, namely all immediate subbases of K_i plus K_i itself. Therefore if the whole $\mathcal{Q}_i$ was taken into account, the ranking given in table 6.3 should be 7.86 / 965. But on average, there were only 80.03 bases in $\mathcal{Q}'_i$, such that the value in table 6.3 is 7.86 / 80.03 instead.

Results are again positive, with significant p-values (to see this, it is important to keep in in mind that the values in column "rank" are the average of observed rankings for all 100 KBs). The preference relations based on a lexicographic ordering generally outperformed the ones based on compliance scores (i.e. on the mean of plausibility scores), and best results were obtained with the relation $\preceq_{lex_{K_i}}$.

6.3 Trimming multiple axioms

Section 6.2 showed that the proposals made in Chapter 4, namely the plausibility score and the four (alternative) preference relations over candidate output bases, may indeed allow for the identification of a single erroneous axiom within an input KB K. Therefore it may be interesting to extend this approach to the identification of (a) set(s) of erroneous axioms in K. The task consists in selecting (an) optimal

subbase(s) of K wrt linguistic evidence. In other words, if $\preceq$ is one of the four abovementioned preference relations, the problem in theory consists in computing $\max_{\preceq} 2^K$.

This section shows that there is no obvious algorithmic solution to this problem. Section 6.3.1 explains why for at least two of these four preference relations, computing all preferred subbases of K may be unfeasible in practice. Section 6.3.2 proposes two algorithms to address this task. The first one is a simple greedy alternative, and is evaluated with real and automatically degraded datasets. The second algorithm exactly computes the optimal subbases of K wrt one of the four preference relations, namely $\preceq_{lex_K}$, but its execution may be prohibitive in practice. It is shown in particular that as far as worst-case complexity is concerned, the problem of computing all optimal subbases of K wrt $\preceq_{lex_K}$ can be reduced to computing all base remainders (defined in Chapter 3 Section 3.6.2.1) for each $\psi \in \Psi_K$.

6.3.1 Lack of an obvious algorithmic solution

The four preference relations over $\mathcal{Q}$ defined in Chapter 4 Section 4.2.5 are different ways of ranking candidate subbases according to the plausibility of their respective consequences. A candidate output base $Q \in \mathcal{Q}$ is evaluated based on the plausibility scores computed for each $\psi \in \Psi_Q$. But if $|\mathcal{Q}|$ is large (for instance here $|\mathcal{Q}| = 2^k$), finding the optimal elements of $\mathcal{Q}$ according to one of these preference relations may be non-trivial.

A first reason is that what is evaluated here is the plausibility of consequences, and not of axioms. So even if one could identify a subset Ψ' of Ψ_K which is optimal, there may not even exist a subbase Q of K such that $\Psi_Q = \Psi'$. For instance, let us assume that $\Psi_K = \{$`Bank`$(Caixa\ Bank)$, `Bank`$(Sina\ Bank)$, `Bank`$(Barclays)$,

$\texttt{Bank}(\textit{Margaret Atwood})\}$. It may be possible to identify that the best possible subset of Ψ_K wrt to the linguistic input is $\Psi'_K = \Psi_K \setminus \{\texttt{Bank}(\textit{Margaret Atwood})\}$. But there may be no subbase Q of K such that $\Psi_Q = \Psi'$. And clearly, if $\mathcal{Q} = 2^K$ (or even if $|\mathcal{Q}|$ is simply large), computing $\mathcal{Q}$ as well as Ψ_Q for each $Q \in \mathcal{Q}$ is not a realistic option.

Monotonicity seems like a relatively obvious candidate property in order to optimize the exploration of $\mathcal{Q}$, i.e the fact that if $Q_1 \subseteq Q_2$, then $\mathrm{Cn}(Q_1) \subseteq \mathrm{Cn}(Q_2)$, and therefore also $\Psi_{Q_1} \subseteq \Psi_{Q_2}$. Intuitively, if some properties are verified by the plausibility scores of (some of) the formulas of Ψ_Q, this should spare the need for an exploration of either subsets or supersets of Q.

But for the preference relations $\preceq_{\mathrm{comp}}$ and $\preceq_{lex}$, the plausibility score $\mathrm{sc}_Q(\psi)$ for each $\psi \in \Psi_Q$ is computed wrt to Q. So it doesn't hold in general that $\mathrm{sc}_{Q_1}(\psi) = \mathrm{sc}_{Q_2}(\psi)$, because the support set for ψ in Q_1 may differ from its support set in Q_2. In particular, it may be the case that $Q_1 \subseteq Q_2$ but $\mathrm{sc}_{Q_1}(\psi) > \mathrm{sc}_{Q_2}(\psi)$, such that for a given $Q \in \mathcal{Q}$, monotonicity does not guarantee that no subset or superset of Q needs to be investigated.

This is why the preference relations $\preceq_{\mathrm{comp}_K}$ and $\preceq_{lex_K}$ were introduced. For these, the plausibility scores are computed only once wrt K, and not for each $Q \in \mathcal{Q}$. This is arguably less satisfying, in that a candidate subbase Q is not evaluated based on its own compliance to linguistic evidence (except for the case where $\Psi_Q = \Psi_K$). But this is also more amenable to optimizations. In particular, the algorithm described in Section 6.3.2.2 computes $\preceq_{lex_K}$ by iteratively refining a family $\mathcal{Q} \subseteq 2^K$ such that at any step of the execution, if $Q \in \max_{\preceq_{lex_K}} 2^K$, then there is a $Q' \in \mathcal{Q}$ such that $Q \subseteq Q'$.

But monotonicity may also in some cases be exploited for $\preceq_{\mathrm{comp}_K}$. A simple

example is the case of a base Q_1 such that $\max_{\psi \in \Psi_{Q_1}} \text{sc}_K(\psi) < \text{comp}_K(Q_2)$, for some already evaluated base $Q_2 \subseteq K$. Then for all $Q'_1 \subseteq Q_1$, $Q'_1 \prec_{\text{comp}_K} Q_2$ must hold, and therefore no subbase of Q_1 can be optimal wrt $\preceq_{\text{comp}_K}$.

6.3.2 Algorithms

This section shows how the four preference relations defined in Chapter 4 Section 4.2.3 can be concretely used to select some set(s) of axioms of an input KB K to be preferably discarded when the family of candidate output bases is (in theory) 2^K.

Section 6.3.2.1 presents a greedy approach which consists in selecting optimal immediate subbases only, but iteratively, and as such can be viewed as a repeated application of the procedure evaluated in Section 6.2. Section 6.3.2.2 on the other hand provides an algorithm which compute exactly the subbases of K which are maximal wrt the preference relation $\preceq_{lex_K}$ defined in Chapter 4 definition 4.2.5. The complexity of this problem is also shown to be identical in the worst case to the complexity of computing all base remainders for each consequence with a plausibility score, i.e. for each $\psi \in \Psi_K$. The first algorithm is evaluated with real and automatically degraded (but consistent) KBs, whereas the applicability of the second algorithms is discussed after its description.

6.3.2.1 Iterated trimming

6.3.2.1.1 Procedure This section describes a very simple approach to select a a subset of 2^K based any of the four preference relations defined in Chapter 4 Section 4.2.3. A family $\mathcal{Q}$ of subsets of K is initialized with $\{K\}$, and iteratively updated. Let $\mathcal{Q}^0, .., \mathcal{Q}^n$ designate the successive states of $\mathcal{Q}$ after each iterative update, with $\mathcal{Q}^0 = \{K\}$, and let $\preceq$ designate any of the four preference relations over 2^K. For

each $0 \leq i \leq n$, let $\mathcal{S}^i$ designate the family of all immediate subbases of any $Q \in \mathcal{Q}^i$, i.e. $\mathcal{S}^i = \bigcup_{Q \in \mathcal{Q}} \{Q \setminus \{\phi\} \mid \phi \in Q\}$. If there is at least one $S \in \mathcal{S}^i$ such that $Q \prec S$ for some $Q \in \mathcal{Q}_i$, then $\mathcal{Q}$ is replaced by all preferred elements of $\mathcal{S}^i$ wrt $\preceq$, i.e. $\mathcal{Q}^{i+1} = \max_{\preceq} \mathcal{S}^i$. If there is no $S \in \mathcal{S}^i$ satisfying this condition, the algorithm terminates. An obvious property of this strategy is the following one:

Proposition 6.3.2.1. For each $0 \leq i \leq n$, for all $Q \in \mathcal{Q}^i$, $|Q| = |K| - i$.

So in practice, another alternative termination condition may be set by bounding the number of axioms to be discarded, i.e. by setting a maximal number m of inductive steps, given as a parameter. If n is important, this is arguably a more realistic debugging scenario. For instance, if $n > \frac{|K|}{2}$, discarding more than a half of K is probably not a desirable output.

The procedure is described by algorithm 1. The output of the process is the variable $\mathcal{O}$, which corresponds to $\mathcal{Q}^n$ if $n \leq m$, and $\mathcal{Q}^m$ otherwise. The function SelectRandom(X), lines 7 and 8, returns a random element of X.

An straightforward but interesting property of algorithm 1 is the following one:

Proposition 6.3.2.2. If $n > 0$, for each $0 \leq i < n$, if $Q_1 \in \mathcal{Q}^i$ and $Q_2 \in \mathcal{Q}^{i+1}$, then $Q_1 \prec Q_2$

This property holds because all four preference relations are total preorders, i.e. intuitively rankings. A consequence of property 6.3.2.2 and the definitions of these preference relations is that if $S \in \mathcal{S}^i$, $Q \in \mathcal{Q}^i$ such that $S \subset Q$ and S and Q are equally reliable wrt to the linguistic input, then S will not be retained as a candidate subbase in $\mathcal{Q}^{i+1}$, which can be viewed as a (weak) form of minimal syntactic information loss.

Algorithm 1 Iterative trimming

$\mathcal{Q} \leftarrow \{K\}$
$\mathcal{O} \leftarrow \{K\}$
$i \leftarrow 0$
while $\mathcal{Q} \neq \emptyset$ and $i \leq m$ **do**
    $\mathcal{Q} \leftarrow \emptyset$
    $\mathcal{S} \leftarrow \bigcup_{Q \in \mathcal{Q}} \{Q \setminus \{\phi\} \mid \phi \in Q\}$
    $S \leftarrow \text{SELECTRANDOM}(\max_{\preceq} \mathcal{S})$
    $Q \leftarrow \text{SELECTRANDOM}(\mathcal{Q})$
    if $Q \prec S$ **then**
        $\mathcal{Q} \leftarrow \max_{\preceq} \mathcal{S}$
    end if
    if $\mathcal{Q} \neq \emptyset$ **then**
        $\mathcal{O} \leftarrow \mathcal{Q}$
    end if
    $i \leftarrow i + 1$
end while

6.3.2.1.2 Evaluation A simplified version of algorithm 1 was implemented in order to reduce the manual cost of the evaluation. It updates a single subbase Q throughout the execution, instead of a family of subbases, initiated with $Q^0 = K$. At each iterative step, a unique $S \in \max_{\preceq} \mathcal{S}^i$ is selected (non-deterministically if $|\max_{\preceq} \mathcal{S}^i| > 1$), and if $Q^i \prec S$, then $Q^{i+1} = S$, otherwise the algorithm terminates. As an alternative termination condition, a maximal number of $m = 20$ axioms to discard was used as a parameter. In practice, for all experiments, the effective termination condition was $i = 20$, i.e. 20 axioms could actually be discarded without earlier termination, or in other words, the situation where $S \preceq Q^i$ for some $S \in \mathcal{S}^i$ was never encountered.

For each $S \in \mathcal{S}^i$, the set of consequences of S for which plausibility scores were computed was Ψ_S, as defined in Section 6.1.2. The datasets used for this evaluation are K_1^{DBP} and $K_{fisheries}$, both described in Chapter 5. As a reminder, if $Q \subseteq K$ is the

output of the process, this means that each $\phi \in K \setminus Q$ is (automatically) designated as potentially erroneous.

For K_1^{DBP}, the evaluation consisted in manually verifying whether a discarded axiom ϕ was actually erroneous, i.e. whether the understanding of some element of the signature of ϕ (individual, atomic concept or role) was incompatible with its overall understanding within K. For instance, the axiom `director`(*Museum Of The Rockies, Smithsonian Institution*) was considered erroneous because, most of the time within K_1^{DBP}, the individual *Smithsonian Institution* is understood as an organization, whereas the DL role `director` is understood as ranging over human beings. Only precision was evaluated, i.e. the proportion of actually erroneous axioms within the automatically discarded ones. For recall, one would need to evaluate the proportion of erroneous axioms of K_1^{DBP} which were automatically discarded. But this requires the prior manual identification of all erroneous axioms within K_1^{DBP} beforehand, which is unrealistic for such a large dataset (> 8000 axioms). This motivated the second evaluation procedure, based on an automatically degraded dataset, which will be presented after this one.

The results are presented in table 6.4. Columns "val." and "prec." respectively give the number and proportion of axioms manually identified as actually erroneous among the 20 discarded ones. Up to 55 % of these automatically discarded axioms were considered as actually erroneous, which is encouraging, because one can reasonably expect the proportion of erroneous within K_1^{DBP} to be considerably lower.

The second experiment is based on an automatically degraded version of $K_{\mathit{fisheries}}$, in order to obtain a more objective (although arguably artificial) evaluation. 20 axioms were randomly generated following the procedure described in Chapter 5 Section 5.2.4, and algorithm 1 was evaluated based on its ability to discard exactly these 20 axioms among the 963 + 20 axioms of the degraded KB. Results are given in table

		val.	prec.
NPS	$\preceq_{\text{comp}}$	7	0.35
	$\preceq_{lex}$	3	0.15
LP	$\preceq_{\text{comp}}$	11	0.55
	$\preceq_{lex}$	3	0.15

Table 6.4: Actually erroneous axioms among the 20 first discarded ones in K_1^{DBP}

		val.	prec. & rec.	p-val (prop. test)
NPS	$\preceq_{\text{comp}}$	9	0.45	< 0.001
	$\preceq_{\text{comp}_K}$	9	0.45	< 0.001
	$\preceq_{lex}$	3	0.15	< 0.002
	$\preceq_{lex_K}$	9	0.45	< 0.001
LP	$\preceq_{\text{comp}}$	10	0.5	< 0.001
	$\preceq_{\text{comp}_K}$	10	0.5	< 0.001
	$\preceq_{lex}$	5	0.25	< 0.001
	$\preceq_{lex_K}$	10	0.5	< 0.001

Table 6.5: Randomly generated statements among the 20 automatically discarded ones for the degraded version of $K_{fisheries}$

6.5, and the values are the number and proportion of randomly generated axioms among the 20 discarded ones. Because the numbers of generated and discarded axioms are identical (20), precision and recall here are the same value, therefore column "prec./rec." estimates both precision and recall. Results are very satisfying, with up to 10 out of the 20 random axioms automatically spotted out of the 983 candidates.

6.3.2.2 Exact lexicographic trimming

This section presents an algorithm which, given an input KB K and a linguistic corpus, yields all subbases of K which are maximal wrt the preference relation $\preceq_{lex_K}$, defined in Chapter 4 definition 4.2.5. The procedure is relatively complex, therefore a non-optimized version will be first presented for readability in Section 6.3.2.2.1. The cost, optimizations and applicability will be discussed in sections 6.3.2.2.2 and

6.3.2.2.3. In particular, it will be shown that for a fixed $|\Psi_K|$, this procedure is optimal (even in its non-optimized version) as far as worst-case complexity is concerned, but may still be prohibitive in practice. Similarly to algorithm 1, running the procedure until termination may not be desirable though, and therefore the possibility to end the execution before termination will also be discussed.

6.3.2.2.1 Procedure As a reminder, Ψ_K designates the set of consequences of K for which a plausibility score could be computed, and for each $\psi \in \Psi_K$, $\mathrm{sc}_K(\psi)$ designates the plausibility score of ψ wrt K. Let $\preceq_{\mathrm{sc}_K}$ be the total preorder over Ψ_K defined by:

Definition 6.3.2.1. $\psi_1 \preceq_{\mathrm{sc}_K} \psi_2$ iff $\mathrm{sc}_K(\psi_2) \leq \mathrm{sc}_K(\psi_2)$.

According to the notation introduced in Chapter 2 Section 2.2, the ordered partition of Ψ_K defined by $\preceq_{\mathrm{sc}_K}$ should be designated with $(\Psi_K)_1^{\preceq_{\mathrm{sc}_K}}, .., (\Psi_K)_m^{\preceq_{\mathrm{sc}_K}}$, i.e. $\mathrm{sc}_K(\psi_1) \leq \mathrm{sc}_K(\psi_2)$ iff there are $1 \leq i \leq j \leq m$ such that $\psi_1 \in (\Psi_K)_i^{\preceq_{\mathrm{sc}_K}}$ and $\psi_2 \in (\Psi_K)_j^{\preceq_{\mathrm{sc}_K}}$. A simplified notation will be adopted in this section: $(\Psi_K)_1^{\preceq_{\mathrm{sc}_K}}, .., (\Psi_K)_m^{\preceq_{\mathrm{sc}_K}}$ will be designated with $\Psi_K^1, .., \Psi_K^m$ instead, and m will always refer to the number of equivalence classes defined by $\preceq_{\mathrm{sc}_K}$ over Ψ_K. In addition, by convention, $\Psi_K^0 = \emptyset$. Similarly, for any $Q \subseteq K$, $(\Psi_Q)_i^{\preceq_{\mathrm{sc}_K}}$ will be denoted with Ψ_Q^i. As a reminder, $\Psi_Q^i = \{\psi \in \Psi_K^i \mid Q \vdash \psi\}$. Finally, the following notation will be used:

Definition 6.3.2.2. $\Psi_Q^{\leq i} = \bigcup_{1 \leq j \leq i} \Psi_Q^j$

Definition 6.3.2.3. $\Psi_Q^{< i} = \bigcup_{1 \leq j < i} \Psi_Q^j$

Definition 6.3.2.4. $\Psi_Q^{\geq i} = \bigcup_{i \leq j \leq m} \Psi_Q^j$

Definition 6.3.2.5. $\Psi_Q^{> i} = \bigcup_{i < j \leq m} \Psi_Q^j$

As a reminder of definition 4.2.5.6, if for all $Q \in \mathcal{Q}$, $\boldsymbol{w}_Q$ is the vector composed of all $\mathrm{sc}_K(\psi)$ for each $\psi \in \Psi_Q$ sorted by increasing value, then $\preceq_{lex_K}$ is based on a standard ascending lexicographic ordering $\preceq_l$ of all $\boldsymbol{w}_Q$, more specifically $Q_1 \preceq_{lex_K} Q_2$ iff $Q_2 \not\prec_{lex_K} Q_1$, and $Q_1 \prec_{lex_K} Q_2$ iff either $\boldsymbol{w}_{Q_1} \prec_l \boldsymbol{w}_{Q_2}$, or $\boldsymbol{w}_{Q_1} =_l \boldsymbol{w}_{Q_2}$ and $Q_1 \subset Q_2$. In the limit case where $m = 1$, i.e. where there is just one equivalence class defined by $\preceq_{\mathrm{sc}_K}$ over Ψ_K (or in other words, if all evaluated consequences of K have the same plausibility score), then trivially, $\max_{\preceq_{lex_K}} 2^K = K$. So the current section will focus instead on the non-trivial case where $m > 1$. The following is an immediate consequence of definition 4.2.5.6:

Proposition 6.3.2.3. $Q \prec_{lex_K} Q_2$ iff either:

(1) for all $1 \leq i \leq m$, $|\Psi^i_{Q_1}| = |\Psi^i_{Q_2}|$, and $Q_1 \subset Q_2$, or

(2) there is a $1 \leq i \leq m$ such that for all $1 \leq j < i$ $|\Psi^j_{Q_1}| = |\Psi^j_{Q_2}|$, and one of the three following conditions holds:

(2.1) $\Psi^{>i}_{Q_1} = \emptyset$ and $\Psi^{>i}_{Q_2} \neq \emptyset$

(2.2) $\Psi^{>i}_{Q_2} \neq \emptyset$, and $|\Psi^i_{Q_2}| < |\Psi^i_{Q_1}|$

(2.3) $\Psi^{>i}_{Q_1} = \emptyset$, $\Psi^{>i}_{Q_2} = \emptyset$, and $|\Psi^i_{Q_2}| > |\Psi^i_{Q_1}|$

To illustrate this, let $\boldsymbol{v}_Q = (|\Psi^1_Q|, \ldots, |\Psi^m_Q|)$, i.e. $\boldsymbol{v}_Q$ is the vector representing the number of consequences of Q in each equivalence class defined by $\preceq_{\mathrm{sc}_K}$ within Ψ_K for any $Q \subseteq K$, and consider the following example:

Ex 6.3.1. $\boldsymbol{v}_K = (2, 4, 2, 3, 2)$
$\boldsymbol{v}_{Q_1} = (0, 2, 0, 1, 0)$
$\boldsymbol{v}_{Q_2} = (0, 2, 2, 2, 0)$

$\boldsymbol{v}_{Q_3} = (0, 2, 2, 1, 0)$

$\boldsymbol{v}_{Q_4} = (0, 1, 0, 0, 0)$

In this example, there are respectively 2, 4, 2, 3 and 2 consequences in Ψ^1_K, Ψ^2_K, Ψ^3_K, Ψ^4_K and Ψ^5_K. The least plausible consequences are the 2 formulas of Ψ^1_K, and the most plausible ones are the 2 formulas of Ψ^5_K. None of Q_1, Q_2, Q_3 and Q_4 has a formula of Ψ^1_K as a consequence, but all of them have higher ranked consequences. Therefore according to property 6.3.2.3 proposition (2.2), they are all strictly preferred to K. Then Q_4 is the base with the smallest number of consequences from Ψ^2_K. So one may be tempted to select Q_4. But applying this strategy systematically would have the undesired property that for any consistent input K, and any $Q \subseteq K$ such that $\Psi_Q \neq \emptyset$, $Q' = \emptyset$ is preferred to Q. Instead, the lexicographic ordering also takes into account the fact that Q_4 has no better ranked consequence, as opposed to Q_1, Q_2 and Q_3, which are therefore all strictly preferred to Q_4, according to property 6.3.2.3 proposition (2.1). To see this more concretely, consider the vector $\boldsymbol{w}_Q$, which contains the ordered plausibility scores of formulas of Ψ_Q, and let us assume that for all $\psi \in \Psi_1$, $\mathrm{sc}_K(\psi) = 0.1$, for all $\psi \in \Psi_2$, $\mathrm{sc}_K(\psi) = 0.2$ etc. Then:

$\boldsymbol{w}_{Q_1} = (0.2, 0.2, 0.4)$,

$\boldsymbol{w}_{Q_2} = (0.2, 0.2, 0.3, 0.3, 0.4, 0.4)$,

$\boldsymbol{w}_{Q_3} = (0.2, 0.2, 0.3, 0.3, 0.4)$,

and $\boldsymbol{w}_{Q_4} = (0.2)$.

So $Q_4 \prec_{l_K} Q_1$, $Q_4 \prec_{l_K} Q_2$ and $Q_4 \prec_{l_K} Q_3$ all hold, such that Q_1, Q_2 and Q_3 are all strictly preferred to Q_4. Then according to proposition 6.3.2.3 property (2.2), Q_1 is strictly preferred to Q_2 and Q_3, because $|\Psi^3_{Q_1}| = 0 < |\Psi^3_{Q_2}| = |\Psi^3_{Q_3}| = 2$, and there is a $3 < j \leq 5$ such that $|\Psi^j_{Q_1}| > 0$ (namely 4). The last interesting case is the ordering of Q_2 and Q_3. $|\Psi^4_{Q_3}| < |\Psi^4_{Q_2}|$ holds, but there is no $4 < j \leq 5$ such that $|\Psi^j_{Q_3}| > 0$, and

therefore Q_2 is preferred to Q_3, according to proposition 6.3.2.3 property (2.3). So the preference relation over these 5 bases is $K \prec_{lex_K} Q_4 \prec_{lex_K} Q_3 \prec_{lex_K} Q_2 \prec_{lex_K} Q_1$.

Here are now a few useful definitions, in order to understand the algorithm. For $1 \leq i \leq m$, let $\mathcal{B}^i$ be defined as follows:

Definition 6.3.2.6. $Q \in \mathcal{B}^i$ iff $\Psi_Q^{\leq i} = \emptyset$ and $\Psi_Q^{>i} \neq \emptyset$

Then t will designate the largest integer such that $\mathcal{B}^t \neq \emptyset$, i.e:

Definition 6.3.2.7. $t = \max_{\leq}\{i \in \{0, 1, .., m-1\} \mid \mathcal{B}^i \neq \emptyset\}$

In the non-trivial case where $\Psi_K \neq \emptyset$, $K \in \mathcal{B}^0$. So there is at least one $0 \leq i < m$ such that $\mathcal{B}^i \neq \emptyset$, and therefore t is defined for non-trivial inputs.

The presentation of the algorithm will be split into two successive phases, for readability, based on the following observation (proven in Section 6.5.1.2):

Proposition 6.3.2.4. If $Q \in \max_{\preceq_{lex_K}} 2^K$, then there is a $B \in \max_{\subseteq} \mathcal{B}^t$ such that $Q \subseteq B$.

Phase 1 yields $\max_{\subseteq} \mathcal{B}^t$ (as well as t). Then if $\mathcal{T} = \bigcup_{B \in \max_{\subseteq} \mathcal{B}^t} 2^B$, phase 2 yields $\max_{\preceq_{lex_K}} \mathcal{T} = \max_{\preceq_{lex_K}} 2^K$.

The notion of base remainder (introduced in Chapter 3 Section 3.6.2.1) will also be needed. If ψ is a consequences of a finite set Γ of axioms, $\mathcal{R}_{\subseteq}(\psi, \Gamma)$ will designate the family of base remainders for ψ in Γ, i.e.:[5]

Definition 6.3.2.8. $R \in \mathcal{R}_{\subseteq}(\psi, \Gamma)$ iff $R \subseteq \Gamma$, $R \nvdash \psi$, and for all $R \subset R' \subseteq \Gamma$, $R' \vdash \psi$.

[5]Equivalently, one could consider the family of diagnoses for ψ in Γ, because each diagnosis is the complement in Γ of a base remainder (see Chapter 3 Section 3.6.2.2). The choice is made here to focus on base remainders instead for readability only. But in practice, an implementation would rather manipulate diagnoses, which tend to be of a smaller size.

Then if Ψ is a set of formulas, $\mathcal{R}^{\vee}_{\subseteq}(\Psi, \Gamma)$ will designate the family of base remainders in Γ for Ψ understood disjunctively, i.e:

Definition 6.3.2.9. Let $\Psi = \{\psi_1, .., \psi_n\}$. Then $R \in \mathcal{R}^{\vee}_{\subseteq}(\Psi, \Gamma)$ iff $\mathrm{Cn}(R) \cap \Psi = \emptyset$, and for all $R \subset R' \subseteq \Gamma$, $\mathrm{Cn}(R') \cap \Psi \neq \emptyset$.

Equivalently, $\mathcal{R}_{\subseteq}(\psi, \Gamma)$ can be viewed as a shortcut for $\mathcal{R}^{\vee}_{\subseteq}(\{\psi\}, \Gamma)$.

Phase 1 is described by algorithm 2. It proceeds by updating two families $\mathcal{Q}$ and $\mathcal{O}$ of subsets of K. Both are initiated with $\{K\}$. Let $\mathcal{Q}^i$ (resp. $\mathcal{O}^i$) designate the state of $\mathcal{Q}$ immediately after incrementing variable i line 17, i.e. after i iterations over the main loop. $\mathcal{O}$ is used to keep in memory the state of $\mathcal{Q}$ during the previous iteration. After termination, i.e. when $\mathcal{Q}^i = \emptyset$, $\mathcal{O}^i = \mathcal{Q}^{i-1}$ is the output of phase 1. The boolean function containsSuperset$(\mathcal{Q}, Q_2)$ line 10 returns true iff there is a $Q \in \mathcal{Q}$ such that $Q_2 \subset Q$. The function discardSubsets$(\mathcal{Q}, Q_2)$ line 11 returns $\mathcal{Q} \setminus \{Q \mid Q \subset Q_2\}$.

After each execution step i, the following property holds (proven in Section 6.5.1.2):

Proposition 6.3.2.5. For $0 \leq i \leq t$, $\mathcal{Q}^i = \max_{\subseteq} \mathcal{B}^i$

This guarantees termination in the worst case when i reaches $m - 1$. because from the definition of $\mathcal{B}$, $\mathcal{B}^m = \emptyset$, such that after m execution steps, $\mathcal{Q} = \emptyset$, and the loop is not entered again.

For correctness, the following is shown to hold in Section 6.5.1.3:

Proposition 6.3.2.6. If $m > 1$, after termination of algorithm 2, $i = t + 1$

So from properties 6.3.2.5 and 6.3.2.6, after termination, $\mathcal{O}^i = \mathcal{Q}^{i-1} = \max_{\subseteq} \mathcal{B}^t$, such that $\mathcal{O}$ is the desired output.

Algorithm 2 Computing $\max_{\subseteq} \mathcal{B}^t$

$\mathcal{Q} \leftarrow \{K\}$
$\mathcal{O} \leftarrow \emptyset$
$i \leftarrow 0$
while $\mathcal{Q} \neq \emptyset$ **do**
    $\mathcal{O} \leftarrow \mathcal{Q}$
    $\mathcal{Q} \leftarrow \emptyset$
    for all $Q_1 \in \mathcal{O}$ **do**
        for all $R \in \mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)$ **do**
            $Q_2 \leftarrow Q_1 \cap R$
            if containsSuperset$(\mathcal{Q}, Q_2) =$ false **then**
                $\mathcal{Q} \leftarrow$ discardSubsets$(\mathcal{Q}, Q_2)$
                $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{Q_2\}$
            end if
        end for
    end for
    $\mathcal{Q} \leftarrow \{Q_2 \in \mathcal{Q} \mid \Psi^{>i+1}_{Q_2} \neq \emptyset\}$
    $i \leftarrow i + 1$
end while

Algorithm 3 Computing $\max_{\preceq_{lex_K}} 2^K$

$\mathcal{Q} \leftarrow \max_{\subseteq} \mathcal{B}^t$
$i \leftarrow t$
while $i < m$ **do**
    $\mathcal{Q} \leftarrow$ Refine$(\mathcal{Q}, i)$
    $i \leftarrow i + 1$
end while
$\mathcal{Q} \leftarrow \max_{\preceq_{lex_K}} \mathcal{Q}$

Algorithm 4 function REFINE

```
1: function REFINE(𝒬, i)
2:     𝒱 = {< R, ψ > | ψ ∈ Ψ_K^{i+1} and R ∈ ℛ_⊆(ψ, K)}
3:     ℋ_def ← ∅
4:     ℋ(K) ← ∅
5:     𝒬_2 ← 𝒬
6:     while 𝒬_2 ≠ ∅ do
7:         𝒪_2 ← 𝒬_2
8:         𝒬_2 ← ∅
9:         for all Q ∈ 𝒬 do
10:            for all R ∈ ℋ_def do
11:                Q_2 ← Q ∩ R
12:                if containsSuperset(𝒬_2, Q_2) = false then
13:                    𝒬_2 ← discardSubsets(𝒬_2, Q_2)
14:                    𝒬_2 ← 𝒬_2 ∪ {Q_2}
15:                end if
16:            end for
17:        end for
18:        𝒬_2 ← {Q_2 ∈ 𝒬_2 | Ψ_{Q_2}^{>i+1} ≠ ∅}
19:        if 𝒬_2 ≠ ∅ then
20:            ℋ = Update(ℋ, 𝒱)
21:        end if
22:    end while
23:    Return 𝒪_2
24: end function
```

Algorithm 5 function UPDATE

```
1: function UPDATE(H, V)
2:     H' ← H
3:     H_def ← ∅
4:     for all R1 ∈ H'_def do
5:         for all <R2, ψ> ∈ V do
6:             if ψ ∉ H'(R1) then
7:                 R3 ← R1 ∩ R2
8:                 if containsSuperset(H_def, R3) = false then
9:                     H ← discardSubsets(H_def, R3)
10:                    H(R3) ← H'(R1) ∪ {ψ}
11:                end if
12:            end if
13:        end for
14:    end for
15:    Return H
16: end function
```

Phase 2 is described by algorithm 3.

A few shortcuts are used in order to keep the notation concise:

- The variable $\mathcal{H}$ designates a lookup table, which associates a $\Psi \subseteq \Psi_K$ to a $Q \subseteq K$. At any step i of the execution, it can be viewed as a partial function from 2^K to $2^{\Psi_K^{i+1}}$. The affectation $\mathcal{H}(Q) \leftarrow \Psi$ indicates that the value for $\mathcal{H}(Q)$ becomes Ψ, whether $\mathcal{H}$ was defined for Q before this operation or not. $\mathcal{H}_{def} \subseteq 2^K$ designates the set of values for which $\mathcal{H}$ is defined. Finally, the affectation $\mathcal{H}_{def} \leftarrow \emptyset$ erases the lookup table, i.e. $\mathcal{H}(Q)$ is undefined for any $Q \in 2^K$ after this operation.

- line 13, discardSubsets$(\mathcal{H}_{def}, Q)$ returns an updated version $\mathcal{H}'$ of $\mathcal{H}$, such that $\mathcal{H}'_{def} \subseteq \mathcal{H}_{def}$, and for any $Q_2 \in \mathcal{H}_{def}$, $\mathcal{H}'(Q_2)$ is undefined if $Q_2 \subset Q$, and $\mathcal{H}'(Q_2) = \mathcal{H}(Q_2)$ otherwise.

Let $h(i)$ be defined inductively as follows:

Definition 6.3.2.10.

- $h(0) = 0$
- Let $M \in \mathcal{M}^i$ iff $M \subseteq K$ and $|\Psi_M^j| = h(j)$ for all $0 \leq j \leq i$.

  If $\bigcup_{M \in \mathcal{M}^i} \Psi_M^{>i+1} \neq \emptyset$, then $h(i+1) = \min_{\leq}\{|\Psi_M^{i+1}| \mid M \in \mathcal{M}^i \text{ and } |\Psi_M^{>i+1}| \neq \emptyset\}$.

In other words, $h(i+1)$ is the smallest integer such that there is some $M \subseteq K$ verifying $\Psi_M^{>i+1} \neq \emptyset$, $|\Psi_M^j| = h(j)$ for all $0 \leq j \leq i$, and $|\Psi_M^{i+1}| = h(i+1)$. Note that there may be a $0 \leq i < m$ such that $h(j)$ is defined for all $0 \leq j \leq i$, but $h(i+1)$ is not. This corresponds to the case where no $M \subseteq K$ verifies $\Psi_Q^{>i+1} \neq \emptyset$ and $|\Psi_Q^j| = h(j)$ for all $0 \leq j \leq i$. Then because h is defined inductively, it is undefined for all $k > i+1$ as well. p will designate the maximal value for which h is defined. Equivalently, p is the smallest value such that for any $M \subseteq K$, if $|\Psi_M^j| = h(j)$ for all $0 \leq j \leq p$, then $\Psi_M^{>p+1} = \emptyset$.

Then $\mathcal{G}^{i.l}$ will designate the family of subsets of K defined by:

Definition 6.3.2.11. For $0 \leq i \leq p$ and $1 \leq l \leq |\Psi_K^i|$, $G \in \mathcal{G}^{i,k}$ iff the three following conditions hold:

- (1) For each $0 \leq j < i$, $|\Psi_G^j| = h(j)$.
- (2) $|\Psi_G^i| \leq k$.
- (3) $\Psi_G^{>i} \neq \emptyset$

The value of variable i will be used to index the successive states of variable $\mathcal{Q}$ immediately after each incrementation of variable i in algorithm 3, such that $\mathcal{Q}^t$ is

the state of $\mathcal{Q}$ after initialization, $\mathcal{Q}^{t+1}$ is the state of $\mathcal{Q}$ after 1 iteration of the main loop, etc. Similarly $\mathcal{Q}_2^{k.l}$ (resp. $\mathcal{H}_{def}^{k.l}$) will designate the state of variable $\mathcal{Q}_2$ (resp. $\mathcal{H}_{def}$) when $i = k$, and immediately after l iterations of the main loop of the function Refine.

For the main loop of algorithm 3, termination is guaranteed by the fact that $t \leq m$, and that i is incremented at each iteration. For the main loop of function Refine, the number of iterations is bounded by $|\Psi_K^{i+1}| + 2$, from the following observation, (proven in Section 6.5.1.5):

Proposition 6.3.2.7. For $t \leq i < m$, $\mathcal{H}_{def}^{i.|\Psi_K^{i+1}|+1} = \emptyset$

In the worst case, during iteration $|\Psi_K^{i+1}|+2$, after the execution of line 8, $\mathcal{Q}_2 = \emptyset$. Then because $\mathcal{H}_{def} = \mathcal{H}_{def}^{i.|\Psi_K^{i+1}|+1} = \emptyset$, the loop line 10 is never entered. So line 18, $\{Q_2 \in \mathcal{Q}_2 \mid \Psi_{Q_2}^{>i+1} \neq \emptyset\} = \{Q_2 \in \emptyset \mid \Psi_{Q_2}^{>i+1} \neq \emptyset\} = \emptyset$, such that $\mathcal{Q}_2 \leftarrow \emptyset$, which corresponds to the termination condition line 6.

For correctness, the following propositions are shown to hold in Section 6.5.1:

Proposition 6.3.2.8. $t \leq p$

Proposition 6.3.2.9. For $t \leq i \leq p$, $\mathcal{Q}^i = \max_{\subseteq} \mathcal{G}^{i,h(i)}$

Proposition 6.3.2.10. For $p < i \leq m$, $\mathcal{Q}^i = \mathcal{Q}^{i-1}$

Proposition 6.3.2.11. $\max_{\preceq_{lex_K}} 2^K = \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$

From the definition of t, $t < m$ must hold. In addition, i is initialized with t, the only exit condition for the main loop is $i = m$, and it has just been shown that the algorithm terminates. So there must be one step in the execution when i reaches m. Then from property 6.3.2.8, and because i is initiated with t and $p < m$ (from the definition of p), there must be one step in the execution when i reaches p. From

property 6.3.2.10, for all the following steps, i.e. for all $p < i \leq m$, $\mathcal{Q}^i = \mathcal{Q}^p$ must hold, such that after exiting the main loop of algorithm 3 when i reaches m, from proposition 6.3.2.9, $\mathcal{Q}^m = \mathcal{Q}^p = \max_{\subseteq} \mathcal{G}^{p,h(p)}$. So line 7 of algorithm 3, $\mathcal{Q}$ takes $\max_{\preceq_{lex_K}} \mathcal{Q}^m = \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$. And from proposition $\mathcal{Q} = \max_{\preceq_{lex_K}} 2^K$, i.e. $\mathcal{Q}$ is the desired output.

6.3.2.2.2 Optimizations As already explained, the above presentation of the procedure remains relatively abstract on purpose, in order to keep the notation concise, and to simplify the proofs of correctness and complexity. In particular, these pseudo-algorithms are poorly optimized, although this does not affect worst-case complexity, which will be discussed in Section 6.3.2.2.3.

A first straightforward optimization of phase 2 consists in stopping iterating over the main loop of algorithm 3 when i reaches $p + 1$. At this stage, from proposition 6.3.2.9, $\mathcal{Q}^i = \mathcal{G}^{p,h(p)}$, and this will be the case for all posterior values of i. In order to ensure that p has been reached, it is sufficient to add an additional exit condition, namely the fact that the main loop of function Refine has been executed only once during the last call to the function. This can only happen in the case where for all $Q \in \mathcal{Q}^i$, $\Psi_Q^{>i+1} = \emptyset$, i.e. $p < i$.

Another optimization concerns the implementation of the verification of $\Psi_Q^{>i+1} \neq \emptyset$, in algorithm 2 line 16, and in function Refine line 18. The number of such verifications during the execution can be potentially high, such that reducing its practical cost may be beneficial. Let Q_1 be the first subset of K for which such a verification is required, with $i = 0$ in this case, i.e. one wants to know whether $\Psi_{Q_1}^{>1} \neq \emptyset$ holds. Then the following strategy may be applied. Formulas of Ψ_K can be ordered by decreasing plausibility, and it can be verified for each $\psi \in \Psi_K^{>1}$, starting with most plausible formulas (i.e. $\psi \in \Psi_K^m$), whether $Q_1 \vdash \psi$ holds, until some

ψ_1 is found which satisfies this condition, or none. If such a ψ_1 is found, one can compute a subset Q'_1 of Q_1 such that $Q'_1 \vdash \psi_1$ and $|Q'_1|$ is preferably small. Q'_1 need not be a justification for ψ_1 in Q_1, but the reduction strategies used for black-box justification finding algorithms (see Chapter 3 Section 3.5.2) can be applied here. Then for another subbase Q_j, if $Q'_k \subseteq Q_j$ for some $1 \leq k < j$ (for instance if $Q'_1 \subseteq Q_j$), this is sufficient to conclude that $\Psi_{Q_j}^{>1} \neq \emptyset$. If there is no Q'_k verifying this condition, the same procedure may be applied in order to find a $\psi_j \in \Psi_K^{>1}$ and a small $Q'_j \subseteq Q_j$ such that $Q'_j \vdash \psi_j$. This holds for $i > 0$ as well, but only provided $\psi_k \in \Psi_K^{>i+1}$, which is the reason why the formulas of Ψ_K were ordered by decreasing plausibility in the first place, in order to maximize the potential reuse of some Q'_k.

6.3.2.2.3 Computational cost and potential use This section will start with a series of observations about the complexity of algorithms 2 and 3, expressed here in the size $|K|$ of the input KB, but regardless of the number $|\Psi_K|$ of consequences of K with a plausibility score. The question of the incidence of $|\Psi_K|$ will only be discussed afterwards. The reason for this choice is that $|\Psi_K|$ does not depend on $|K|$ only, as explained in Chapter 4 Section 4.2.4.1, but also on the syntactic form of consequences of K adopted for the evaluation, otherwise Ψ_K may be infinite.

The input may be any consistent KB K together with a nonempty and finite set Ψ_K of non-tautological consequences of K, and a total preorder $\preceq_{\mathrm{sc}_K}$ over Ψ_K.

Let us assume a procedure which for any input of this type and any $\psi \in \Psi_K$ returns $\mathcal{R}_{\subseteq}(\psi, K)$, and let g be the computational cost of this procedure expressed as a function of $|K|$. Then let us assume any procedure (not necessarily algorithms 2 and 3) which solves the problem adressed by algorithms 2 and 3, i.e. computing $\max \preceq_{lex_K} 2^K$, and let v be the computational cost of this procedure expressed as a function of $|K|$. A first useful remark is the following, which holds regardless of the

underlying DL:

Proposition 6.3.2.12. $v(n) = \Omega(g(n))$

This can be shown with the following example, where $\max_{\preceq_{lex_K}} 2^K = \mathcal{R}_{\subseteq}(\psi_1, K)$:

Ex 6.3.2. $\Psi_K = \{\psi_1, \psi_2\}$

$\psi_1 \prec_{\text{sc}_K} \psi_2$

For all $R \in \mathcal{R}_{\subseteq}(\psi_1, K)$, $R \vdash \psi_2$

Additionally, Section 6.5.2.1 shows that regardless of the underlying DL, the following holds:

Proposition 6.3.2.13. $g(n) = \Omega(c^n)$, with $c > 1$

So g is at least exponential in the worst case, and from proposition 6.3.2.12, this must be the case for v as well.

For more expressive DLs, this may have no incidence on the overall worst-case complexity of the problem, dominated by entailment/satisfiability checks. But for tractable DLs such as $\mathcal{EL}$ or *DL-Lite*, these observations show that computing $\max_{\preceq_{lex_K}} \mathcal{Q}$ (with any algorithm) may still have a cost exponential in $|K|$.

In practice, as explained in Chapter 3 Section 3.5.2, computing $\mathcal{R}_{\subseteq}(\psi, K)$ for a consistent K may be prohibitive, but remains more realistic than computing $\mathcal{R}_{\subseteq}(\bot, K)$ when K is inconsistent. In particular, computing a module of K based on the signature of ψ beforehand may significantly reduce the cost of the operation, as shown empirically by [Hor11] (among others), although worst-case complexity remains identical.

Now let f be the computational cost of the execution of algorithm 2 followed by algorithm 3, expressed as a function of $|K|$. Section 6.5.2.2 shows that regardless of the underlying DL, the following holds:

Proposition 6.3.2.14. $f(n) = O(|\Psi_K| \cdot g(n))$

So if $|\Psi_K|$ is fixed, from propositions 6.3.2.13 and 6.3.2.14, $f(n) = O(g(n))$ holds, which together with proposition 6.3.2.12 implies that worst-case complexity of algorithms 2 and 3 is optimal for the problem they address.

Finally, let $t(n)$ be the maximal value of $|\Psi_K|$ for any input consistent K such that $|K| = n$. If $t(n) = O(c^n)$, i.e. if the number of consequences to be evaluated does not grow more than exponentially with $|K|$, then algorithms 2 and 3 remain optimal in the worst case for the problem they address.

Intuitively, the presence of the value $|\Psi_K|$ in $O(|\Psi_K| \cdot g(n))$ is due to the successive execution steps of algorithms 2 and 3, i.e. the different equivalence classes defined by $\preceq_{\mathrm{sc}_K}$. This raises another problem, which is the question of termination, similarly to the discussion of algorithm 1. One may arguably wonder whether running algorithm 2 until termination (i.e. for all equivalence classes defined by $\preceq_{\mathrm{sc}_K}$) is desirable.

As an (extreme) illustration, let us extend example 6.3.1 in Section 6.3.2.2.1 with an additional candidate subbase Q_5, such that $\boldsymbol{v}_{Q_5} = (0, 0, 0, 0, 1)$. Then Q_5 will be strictly preferred to all other candidate subbases of K in example 6.3.1. But aside from the single formula in $\Psi^5_{Q_5}$, all consequences of Ψ_K will be lost.

An alternative but still intuitive termination condition may be used instead, based on the following observation:

Proposition 6.3.2.15. For $0 \leq i \leq p$, for each $G \in \mathcal{G}^{i,h(i)}$ and any $G \subset Q \subseteq K$, $Q \prec_{lex_K} G$.

In other words, if $G \in \mathcal{G}^{i.h(i)}$, then it is strictly preferred wrt $\prec_{lex_K}$ to any of its supersets in K. In addition, from the definitions of h and p and proposition 6.3.2.15, any $Q \subseteq K$ with $|\Psi^j_Q| < h(j)$ for some $0 \leq j \leq p$ is such that $Q \prec_{lex_K} G$ as well. So

intuitively, $\mathcal{G}^{i.h(i)}$ is an optimal output wrt $\preceq_{lex_K}$ if one focuses on the loss of consequences of the i worst equivalence classes defined by $\preceq_{\text{sc}_K}$ over Ψ_K, or equivalently if all strictly better ranked consequences, i.e. all formulas of $\Psi_K^{>i}$, are merged into a single equivalence class wrt $\preceq_{\text{sc}_K}$. Therefore a relatively straightforward alternative termination condition consists in bounding i beforehand with a parameter $q \leq m$, focusing on the removal of axioms from the q most implausible classes of consequences of K, i.e. from $\Psi_K^1, .., \Psi_K^q$.

6.4 Conclusion

The proposals made in Chapter 4 provided a way to evaluate the compliance of a set of formulas with a linguistic corpus, and, if $\mathcal{Q}$ is the set of candidate output KBs of a debugging process, to rank the elements of $\mathcal{Q}$ based on linguistic evidence. But no concrete indication was given as to how these scores and rankings could be incorporated to an actual debugging process, and therefore these proposals were not evaluated either. The present chapter partially answered both questions, focusing on the case where the input KB K of the debugging process is logically consistent.

Section 6.1 evaluated the accuracy of the linguistic plausibility score defined in Chapter 4. Plausibility scores by themselves do not provide a way to debug a KB, but an isolated evaluation of their accuracy is nonetheless interesting, because they are the core of all other linguistic based debugging strategies presented in this thesis. The evaluation relies on 200 automatically degraded (but consistent) versions of 2 KBs described in Chapter 5. Each of the 2 KBs was extended 100 times with a single random axiom. The evaluation procedure is based on the assumption that new consequences obtained after degrading an input KB with a random axiom are very likely to be nonsensical, when compared to the rest of the KB. The goal of the

evaluation was then to automatically identify such consequences as the least plausible ones among consequences of each degraded KB. The results are significant, i.e. these consequences were generally identified as unlikely to hold if the rest of the degraded KB does, and this result was obtained from linguistic evidence only.

Section 6.2 presented a first evaluation of the incorporation of linguistic evidence to an actual debugging process. It focused on the limited case where one wants to identify a single erroneous axiom to discard from K. The datasets used for this evaluation were the same as the ones used Section 6.1, but the evaluation procedure was more straightforward. This time, the objective was to automatically identify on a linguistic basis that the randomly generated axiom ψ is the least reliable within $K \cup \{\psi\}$, or equivalently, to identify K as best immediate subbase of $K \cup \{\psi\}$. Once again, results are statistically significant.

Section 6.3 focused on the generalization of the approach to the identification of (a) set(s) of erroneous axioms within K. In particular, it showed that the problem is algorithmically complex if the search space is actually 2^K. Section 6.3.2.1 proposed instead a simple greedy approach, evaluated with real and automatically degraded data. The results of these experiments were also significant, but additional evaluations based on different datasets are probably required before drawing conclusions about its applicability. Then Section 6.3.2.2, which is the main contribution of this chapter, addressed the problem of the exact computation of the optimal subbase(s) wrt one of the four preference relations of candidate outputs defined in Chapter 4. A (relatively complex) procedure is provided, as well as complexity bounds, showing that the problem is as hard computationally as computing all base remainders (see Chapter 3 Section 3.6.2.1) for each consequence of K with a plausibility score (assuming the number of such consequences is fixed, or at most exponential in $|K|$). This may turn out to be costly, although running the algorithm only partially is a

potential application scenario.

6.5 Proofs

6.5.1 Section 6.3.2.2

6.5.1.1 Proposition 6.3.2.5

Proposition. If $Q \in \max_{\preceq_{lex_K}} 2^K$, then there is a $B \in \max_{\subseteq} \mathcal{B}^t$ such that $Q \subseteq B$.

Proof. From the definition of t, $\mathcal{B}^t \neq \emptyset$. So there is a $B \in \mathcal{B}^t$, and B verifies $\Psi_B^{\leq t} = \emptyset$, as well as $\Psi_B^{>t} \neq \emptyset$.

Let $Q \in \max_{\preceq_{lex_K}} 2^K$. If $\Psi_Q^{\leq t} \neq \emptyset$, then there is a $1 \leq i \leq t$ such that $|\Psi_Q^j| = |\Psi_B^j| = \emptyset$ for all $1 \leq j < i$, and $|\Psi_B^i| < |\Psi_Q^i|$. And because $\Psi_B^{>t} \neq \emptyset$ and $i \leq t$, $\Psi_B^{>i} \neq \emptyset$, such that from proposition 6.3.2.3 property (2.2), $Q \prec_{lex_K} B$, a contradiction. So $\Psi_Q^{\leq t} = \emptyset$ must hold.

Then if $\Psi_Q^{>t} = \emptyset$, from proposition 6.3.2.3 property (2.1), $Q \prec_{lex_K} B$ too, such that $\Psi_Q^{>t} \neq \emptyset$ must hold as well. So from the definition of $\mathcal{B}$, $Q \in \mathcal{B}^t$. Therefore there must be a maximal element B' of $\mathcal{B}^t$ wrt $\subseteq$ such that $Q \subseteq B'$. □

6.5.1.2 Proposition 6.3.2.5

Proposition. For $0 \leq i \leq t$, $\mathcal{Q}^i = \max_{\subseteq} \mathcal{B}^i$

6.5.1.2.1 Lemmas

Lemma 6.5.1.1. For $0 \leq i < m$, for any $\mathcal{M} \subseteq 2^K$, $Q \in \max_{\subseteq}\{\Gamma \subseteq K \mid \Gamma \in \mathcal{M} \text{ and } \Psi_\Gamma^{>i} \neq \emptyset\}$ iff $Q \in \max_{\subseteq} \mathcal{M}$ and $\Psi_Q^{>i} \neq \emptyset$.

Proof. Let $\mathcal{P} = \{\Gamma \subseteq K \mid \Gamma \in \mathcal{M} \text{ and } \Psi_\Gamma^{>i} \neq \emptyset\}$, and let us start with the left inclusion, i.e. if $Q \in \max_{\subseteq} \mathcal{P}$, then $Q \in \max_{\subseteq} \mathcal{M}$ and $\Psi_Q^{>i} \neq \emptyset$. Let $Q \in \max_{\subseteq} \mathcal{P}$.

Then $\Psi_Q^{>i} \neq \emptyset$. Now let us assume by contradiction that $Q \notin \max_\subseteq \mathcal{M}$. Then there is a $Q_2 \in \mathcal{M}$ such that $Q \subset Q_2$. But by monotonicity, $\Psi_{Q_2}^{>i} \neq \emptyset$ must hold, such that $Q_2 \in \mathcal{P}$, and therefore $Q \notin \max_\subseteq \mathcal{P}$, contradicting the hypothesis.

For the right inclusion, if $Q \in \max_\subseteq \mathcal{M}$ and $\Psi_Q^{>i} \neq \emptyset$, then $Q \in \mathcal{P}$. Let us assume by contradiction that $Q \notin \max_\subseteq \mathcal{P}$. Then there is a $Q_2 \in \mathcal{P}$ such that $Q \subset Q_2$, but because $Q_2 \in \mathcal{P}$, $Q_2 \in \mathcal{M}$, so $Q \notin \max_\subseteq \mathcal{M}$, contradicting the hypothesis. □

Lemma 6.5.1.2. $Q \in \max_\subseteq \mathcal{B}^i$ iff $Q \in \max_\subseteq \{\Gamma \subseteq K \mid \Psi_\Gamma^{\leq i} = \emptyset\}$ and $\Psi_Q^{>i} \neq \emptyset$

Proof. From the definition of $\mathcal{B}$, $Q \in \mathcal{B}^i$ iff $Q \in \{\Gamma \subseteq K \mid \Psi_\Gamma^{\leq i} = \emptyset \text{ and } \Psi_\Gamma^{>i} \neq \emptyset\}$. Then the proof is immediate from lemma 6.5.1.1, replacing $\mathcal{M}$ by $\{\Gamma \subseteq K \mid \Psi_\Gamma^{\leq i} = \emptyset\}$. □

6.5.1.2.2 Main proposition

Proposition. For $0 \leq i \leq t$, $\mathcal{Q}^i = \max_\subseteq \mathcal{B}^i$

Proof. By induction on i. For the base case $i = 0$, from the definition of $\mathcal{B}$, $\mathcal{B}^0 = \max_\subseteq \mathcal{B}^0 = \{K\}$, and $\mathcal{Q}$ is initialized with $\mathcal{Q}^0 = \{K\}$.

For the inductive case, let us assume that $i < t$, and $\mathcal{Q}^i = \max_\subseteq \mathcal{B}^i$ holds by IH. This choice is made (instead of i and $i-1$) in order to follow the notation used in algorithm 2. Variable names (Q_1, Q_2, R) also correspond to the ones used in algorithm 2 whenever possible. From lemma 6.5.1.2, we need to show that $Q \in \mathcal{Q}^{i+1}$, iff $Q \in \max_\subseteq \{\Gamma \subseteq K \mid \Psi_\Gamma^{\leq i+1} = \emptyset\}$ and $\Psi_Q^{>i+1} \neq \emptyset$.

Let us start with the left inclusion, i.e. if $Q \in \mathcal{Q}^{i+1}$, then $Q \in \max_\subseteq \{\Gamma \subseteq K \mid \Psi_\Gamma^{\leq i+1} = \emptyset\}$ and $\Psi_Q^{>i+1} \neq \emptyset$. By contradiction, let us assume that there is a $Q_2 \in \mathcal{Q}^{i+1}$ such that either $Q \notin \max_\subseteq \{\Gamma \subseteq K \mid \Psi_\Gamma^{\leq i+1} = \emptyset\}$, or $\Psi_{Q_2}^{>i+1} = \emptyset$. Line 16 of algorithm 2, Q_2 is retained in $\mathcal{Q}$ only if $\Psi_{Q_2}^{>i+1} \neq \emptyset$, so the only remaining possibility

is $Q_2 \notin \max_{\subseteq}\{\Gamma \subseteq K \mid \Psi_{\Gamma}^{\leq i+1} = \emptyset\}$. Line 9, $Q_2 = Q_1 \cap R$, and by IH, $\Psi_{Q_1}^{\leq i} = \emptyset$, so by monotonicity, $\Psi_{Q_2}^{\leq i} = \emptyset$. And because $R \in \mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{i+1}, K)$, $\Psi_R^{i+1} = \emptyset$ must hold as well, so by monotonicity, $\Psi_{Q_2}^{i+1} = \emptyset$, such that $\Psi_{Q_2}^{\leq i+1} = \Psi_{Q_2}^{\leq i} \cup \Psi_{Q_2}^{i+1} = \emptyset \cup \emptyset = \emptyset$. So the last possibility is the case where $\Psi_{Q_2}^{\leq i+1}$ holds, but Q_2 is not maximal wrt $\subseteq$ within $\{\Gamma \subseteq K \mid \Psi_{\Gamma}^{\leq i+1} = \emptyset\}$, or in other words there is a $Q_2 \subset Q_3$ such that $\Psi_{Q_3}^{\leq i+1} = \emptyset$. If $\Psi_{Q_3}^{\leq i+1} = \emptyset$, then $\Psi_{Q_3}^{\leq i} = \emptyset$ must hold too. Then by monotonicity, because $\Psi_{Q_2}^{>i+1} \neq \emptyset$, $\Psi_{Q_3}^{>i+1} \neq \emptyset$, and therefore $\Psi_{Q_3}^{>i} \neq \emptyset$ must hold as well. So from the definition of $\mathcal{B}$, we have $Q_3 \in \mathcal{B}^i$, such that there must be a $Q_4 \in \max_{\subseteq} \mathcal{B}^i$ with $Q_3 \subseteq Q_4$, and by IH, $Q_4 \in \mathcal{Q}^i$. Then because $\Psi_{Q_3}^{\leq i+1} = \emptyset$, there is an $R \in \mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{i+1}, K)$ such that $Q_3 \subseteq R$. Now let $Q_5 = Q_4 \cap R$, and let $\mathcal{M}$ designate $\{Q \cap R \mid \ < Q, R > \in \mathcal{Q}^i \times \mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{i+1}, K)\}$. Then $Q_5 \in \mathcal{M}$, and immediately from algorithm 2, $\mathcal{Q}^{i+1} \subseteq \max_{\subseteq} \mathcal{M}$. But $Q_2 \subset Q_3$ and $Q_3 \subseteq Q_5$, such that $Q_2 \subset Q_5$ must hold. So $Q_2 \notin \max_{\subseteq} \mathcal{M}$, and therefore $Q_2 \notin \mathcal{Q}^{i+1}$, which contradicts the hypothesis.

Let us continue with the right inclusion, i.e. if $Q \in \max_{\subseteq}\{\Gamma \subseteq K \mid \Psi_{\Gamma}^{\leq i+1} = \emptyset\}$ and $\Psi_Q^{>i+1} \neq \emptyset$, then $Q \in \mathcal{Q}^{i+1}$. Let us assume by contradiction that there is a Q_2 such that $Q_2 \in \max_{\subseteq}\{\Gamma \subseteq K \mid \Psi_{\Gamma}^{\leq i+1} = \emptyset\}$ and $\Psi_{Q_2}^{>i+1} \neq \emptyset$, but $Q_2 \notin \mathcal{Q}^{i+1}$. Because $\Psi_{Q_2}^{\leq i+1} = \emptyset$, $\Psi_{Q_2}^{\leq i} = \emptyset$ must hold, so there is a maximal subset Q_1 of K such that $\Psi_{Q_1}^{\leq i} = \emptyset$ and $Q_2 \subseteq Q_1$, i.e $Q_1 \in \max_{\subseteq}\{\Gamma \subseteq K \mid \Psi_{\Gamma}^{\leq i} = \emptyset\}$. And because $\Psi_{Q_2}^{>i+1} \neq \emptyset$, $\Psi_{Q_2}^{>i} \neq \emptyset$, so by monotonicity $\Psi_{Q_1}^{>i} \neq \emptyset$, such that from lemma 6.5.1.2, $Q_1 \in \mathcal{B}^i$, and by IH, $Q_1 \in \mathcal{Q}^i$. Then because $\Psi_{Q_2}^{i+1} = \emptyset$, there is an $R \in \mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{i+1}, K)$ such that $Q_2 \subseteq R$. So there must be a pair $< Q_1, R > \in \mathcal{Q}^i \times \mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{i+1}, K)$ such that $Q_2 \subseteq Q_1 \cap R$. And line 7 to 9 of algorithm 2, all possible intersections for a pair in $\mathcal{Q}^i \times \mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{i+1}, K)$ are generated, so Q_2 must be one these. If $Q_2 \in \mathcal{Q}$ immediately before the execution of line 16, then Q_2 is retained in $\mathcal{Q}$, because $\Psi_{Q_2}^{>i+1} \neq \emptyset$. So the only remaining possibility for $Q_2 \notin \mathcal{Q}^{i+1}$ is lines 10 to 13: Q_2 may be discarded only if there is a Q_3 such that $Q_2 \subset Q_3$ and $Q_3 = Q_4 \cap R_2$ for some $< Q_4, R_2 > \in$

$\mathcal{Q}^i \times \mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)$. By IH, $\mathcal{Q}^i = \mathcal{B}^i$, such that $Q_4 \in \mathcal{B}^i$, and therefore $\Psi^{\leq i}_{Q_4} = \emptyset$, so by monotonicity $\Psi^{\leq i}_{Q_3} = \emptyset$. And from the definition of a base remainder, $\Psi^{i+1}_{R_2} = \emptyset$, so by monotonicity $\Psi^{i+1}_{Q_3} = \emptyset$. Therefore $\Psi^{\leq i+1}_{Q_3} = \Psi^{\leq i}_{Q_3} \cup \Psi^{i}_{Q_3} = \emptyset$. So because $Q_2 \subset Q_3$, $Q_2 \notin \max_{\subseteq}\{\Gamma \subseteq K \mid \Psi^{\leq i+1}_{\Gamma} = \emptyset\}$, which contradicts the hypothesis. □

6.5.1.3 Proposition 6.3.2.6

Proposition. If $K \neq \emptyset$ and $m > 1$, after termination of algorithm 2, $i = t + 1$

6.5.1.3.1 Lemmas

Lemma 6.5.1.3. For $0 \leq i \leq m - 1$, if $\mathcal{B}^i = \emptyset$, then $\mathcal{B}^{i+1} = \emptyset$

Proof. Take any $Q \subseteq K$. If $\mathcal{B}^i = \emptyset$, then $Q \notin \mathcal{B}^i$, so from the definition of $\mathcal{B}$, either $\Psi^{\leq i}_{Q} \neq \emptyset$ or $\Psi^{>i}_{Q} = \emptyset$ must hold. In the former case, $\Psi^{\leq i}_{Q} \neq \emptyset$ iff there is a $k \leq i$ such that $\Psi^{k}_{Q} \neq \emptyset$, and because $k \leq i+1$, $\Psi^{\leq i+1}_{Q} \neq \emptyset$ must hold too, such that $Q \notin \mathcal{B}i+1$. In the latter case, $\Psi^{>i}_{Q} = \emptyset$ iff there is no $k > i$ such that $\Psi^{k}_{Q} \neq \emptyset$. Because $i \leq i+1$, there is no $k > i+1$ such that $\Psi^{k}_{Q} \neq \emptyset$ either, and therefore $Q \notin \mathcal{B}^{i+1}$. □

6.5.1.3.2 Main proposition

Proposition. If $m > 1$, after termination of algorithm 2, $i = t + 1$.

Proof. As a reminder, if $m > 1$, then t is defined, and $t \geq 0$ must hold.

From the definition of t, $B^t \neq \emptyset$ must hold, and from the contraposition of lemma 6.5.1.3.1, for all $1 \leq j \leq t$, $B^j \neq \emptyset$. But from proposition 6.3.2.5, $\mathcal{Q}^j = \max_{\subseteq} \mathcal{B}^j$, so $\mathcal{Q}^j \neq \emptyset$, such that the main loop is iterated over at least t times. Then from the definition of t, $\mathcal{B}^{t+1} = \emptyset$, and so from proposition 6.3.2.5 $\mathcal{Q}^{t+1} = \mathcal{B}^{t+1} = \emptyset$, and the main loop is exited. □

6.5.1.4 Proposition 6.3.2.8

Proposition. $t \leq p$

Proof. From the definition of t, there is a $Q \subseteq K$ such that $\Psi_Q^{\geq t} \neq \emptyset$, and for $0 \leq i \leq t$, $\Psi_Q^i = \emptyset$. So immediately from the definition of h, for $0 \leq i \leq t$, $h(i) = \emptyset$, therefore h is defined for t, and because p is the largest integer for which h is defined, $t \leq p$. □

6.5.1.5 Proposition 6.3.2.7

Proposition. For $t \leq i < m$, $\mathcal{H}_{def}^{i.|\Psi_K^{i+1}|+1} = \emptyset$

6.5.1.5.1 Lemmas

Lemma 6.5.1.4. For $t \leq i < m$ and $0 \leq l \leq |\Psi_K^{i+1}|$, for each $R \in \mathcal{H}_{def}^{i.l}$, $\mathcal{H}^{i.l}(R) \subseteq \Psi_K^{i+1}$

Proof. Take any $t \leq i < m$. By induction on l. If $l = 0$, i.e. before the first execution of the main loop of function Refine, then $\mathcal{H}^{i.0}$ is defined for K only, i.e. $\mathcal{H}_{def}^{i.0} = \{K\}$, and $\mathcal{H}^{i.0}(K) = \emptyset \subseteq \Psi_K^{i+1}$.

For the inductive case, line 10 of function Update, if $R_3 \in \mathcal{H}_{def}^{i.l}$, then $\mathcal{H}^{i.l}(R_3) = \mathcal{H}^{i.l-1}(R_1) \cup \{\psi\}$ for some $R_1 \in \mathcal{H}_{def}^{i.l-1}$ and some ψ such that there is an R_2 with $< R_2, \psi > \in \mathcal{V}$. By IH, $\mathcal{H}^{i.l-1}(R_1) \subseteq \Psi_K^{i+1}$, and from the initialization of $\mathcal{V}$ line 2 of function Refine, $\psi \in \Psi_K^{i+1}$, so $\mathcal{H}^{i.l-1}(R_1) \cup \{\psi\} = \mathcal{H}^{i.l}(R_3) \subseteq \Psi_K^{i+1}$ □

Lemma 6.5.1.5. For $t \leq i < m$ and $0 \leq l \leq |\Psi_K^{i+1}|$, for each $R \in \mathcal{H}_{def}^{i.l}$, $|\mathcal{H}^{i.l}(R)| = l$

Proof. Immediate from the description of function Refine, by induction on l. □

6.5.1.5.2 Main proposition

Proposition. For $t \leq i < m$, $\mathcal{H}_{def}^{i.|\Psi_K^{i+1}|+1} = \emptyset$

Proof. Take any $t \leq i < m$, and let us assume that $l = |\Psi_K^{i+1}| + 1$, i.e. the main loop of function REFINE has been iterated over $|\Psi_K^{i+1}| + 1$ times (note that this may not be the case for all i, i.e the function may return after less than $|\Psi_K^{i+1}| + 1$ iterations). Then during the previous iteration, i.e. during iteration $|\Psi_K^{i+1}|$, the first argument of the call to function UPDATE line 20 is $\mathcal{H}_{def}^{i.|\Psi_K^{i+1}|}$. Take any $R_1 \in \mathcal{H}_{def}^{i.|\Psi_K^{i+1}|}$. From lemma 6.5.1.4, $\mathcal{H}^{i.|\Psi_K^{i+1}|}(R_1) \subseteq \Psi_K^{i+1}$, and from lemma 6.5.1.5, $|\mathcal{H}^{i.|\Psi_K^{i+1}|}(R_1)| = |\Psi_K^{i+1}|$, such that $\mathcal{H}^{i.|\Psi_K^{i+1}|}(R_1) = \Psi_K^{i+1}$ must hold. In addition, if $< R_2, \psi > \in \mathcal{V}$, then from the initialization of $\mathcal{V}$ line 2 of function REFINE, $\psi \in \Psi_K^{i+1}$. So line 6 of function UPDATE, because $\mathcal{H}' = \mathcal{H}^{i.|\Psi_K^{i+1}|}$, the condition $\psi \notin \mathcal{H}'(R_1)$ is never verified, such that line 10 is never executed, i.e. no R_3 is added to $\mathcal{H}_{def}$, and therefore at the end of the execution of iteration $|\Psi_K^{i+1}| + 1$, $\mathcal{H}_{def}^{|\Psi_K^{i+1}|+1} = \emptyset$. □

6.5.1.6 Proposition 6.3.2.9

Proposition. For $t \leq i \leq p$, $\mathcal{Q}^i = \max_{\subseteq} \mathcal{G}^{i,h(i)}$

6.5.1.6.1 Lemmas

Lemma 6.5.1.6. For $t \leq i < m$ and $0 \leq l \leq |\Psi_K^{i+1}|$, if $R \in \mathcal{H}_{def}^{i.l}$, then $\text{Cn}(R) \cap \mathcal{H}^{i.l}(R) = \emptyset$

Proof. By induction on l. For the base case $l = 0$, i.e. before the first execution of the main loop of function REFINE, $\mathcal{H}_{def}^{i.0} = \{K\}$, and $\mathcal{H}^{i.0}(K) = \emptyset$, so trivially, for all $R \in \mathcal{H}_{def}^{i.0}$, $\text{Cn}(R) \cap \mathcal{H}^{i.0}(R) = \emptyset$.

For the induction step, after the l^{th} call to the function UPDATE, from lines 4, 5 and 7 of function UPDATE, if $R_3 \in \mathcal{H}_{def}^{i.l}$, then $R_3 = R_1 \cap R_2$, such that:

- $R_1 \in \mathcal{H}_{def}^{i.l-1}$
- there is a $\psi \in \Psi_K^{i+1}$ such that $< R_2, \psi > \in \mathcal{V}^i$.
- $\mathcal{H}^{i.l}(R_3) = \mathcal{H}^{i.l-1}(R_1) \cup \{\psi\}$

By IH, $\mathrm{Cn}(R_1) \cap \mathcal{H}^{i.l-1}(R_1) = \emptyset$. And because $< R_2, \psi > \in \mathcal{V}$, $R_2 \in \mathcal{R}_{\subseteq}(\psi, K)$, so $R_2 \nvdash \psi$. Therefore by monotonicity, $\mathrm{Cn}(R_3) \cap (\mathcal{H}^{i.l-1}(R_1) \cup \{\psi\}) = \mathrm{Cn}(R_3) \cap \mathcal{H}^{i.l}(R_3) = \emptyset$. □

Lemma 6.5.1.7. For $t \leq i \leq p$ and $0 \leq l \leq |\Psi_K^{i+1}|$, $\mathcal{H}_{def}^{i.l} = \max_{\subseteq}\{Q \subseteq K \mid |\Psi_Q^{i+1}| \leq |\Psi_K^{i+1}| - l\}$

Proof. By induction on l. For the base case $l = 0$, trivially, $|\Psi_K^{i+1}| \leq |\Psi_K^{i+1}| - 0$, and $\{K\} = \max_{\subseteq} 2^K$, so $\max_{\subseteq}\{Q \subseteq K \mid |\Psi_Q^{i+1}| \leq |\Psi_K^{i+1}|\} = \{K\} = \mathcal{H}_{def}^{i.0}$.

For the inductive case, during the l^{th} execution of the main loop of function Refine, let $\mathcal{N} = \{R_1 \cap R_2 \mid R_1 \in \mathcal{H}_{def}^{i.l-1}$, and there is an $< R_2, \psi > \in \mathcal{V}$ such that $\psi \notin \mathcal{H}^{i.l-1}(R_1)\}$. Then immediately from the description of the algorithm, the call to the function Update line 20 returns $\mathcal{H}^{i.l}$ such that $\mathcal{H}_{def}^{i.l} = \max_{\subseteq} \mathcal{N}$.

Let $\mathcal{M} = \{Q \subseteq K \mid |\Psi_Q^{i+1}| \leq |\Psi_K^{i+1}| - l\}$, and let us start with the left inclusion, i.e. $\mathcal{H}_{def}^{i.l} \subseteq \max_{\subseteq} \mathcal{M}$. Let $R \in \mathcal{H}_{def}^{i.l} = \max_{\subseteq} \mathcal{N}$. From lemma 6.5.1.4, $\mathcal{H}^{i.l}(R) \subseteq \Psi_K^{i+1}$. From lemma 6.5.1.5, $|\mathcal{H}^{i.l}(R)| = l$. And from lemma 6.5.1.6, $\mathrm{Cn}(R) \cap \mathcal{H}^{i.l}(R) = \emptyset$. So there are at least l formulas in $\Psi_K^{i+1} \setminus \mathrm{Cn}(R)$, i.e. $|\Psi_R^{i+1}| \leq |\Psi_K^{i+1}| - l$, or equivalently, $R \in \mathcal{M}$. Now let us assume by contradiction that $R \notin \max_{\subseteq} \mathcal{M}$. Then there must be an $R_2 \in \mathcal{M}$ such that $R \subset R_2$. Because $R_2 \in \mathcal{M}$, $|\Psi_{R_2}^{i+1}| \leq |\Psi_K^{i+1}| - l$. So $|\Psi_{R_2}^{i+1}| \leq (|\Psi_K^{i+1}| - l) + 1 = |\Psi_K^{i+1}| - (l-1)$ also holds, such that $R_2 \in \{Q \subseteq K \mid |\Psi_Q^{i+1}| \leq |\Psi_K^{i+1}| - (l-1)\}$. Therefore by IH, there must be an $R_3 \in \mathcal{H}_{def}^{i.l-1}$ such that $R_2 \subseteq R_3$. From lemma 6.5.1.6, $\mathrm{Cn}(R_3) \cap \mathcal{H}^{i.l-1}(R_3) = \emptyset$. So by monotonicity $\mathrm{Cn}(R_2) \cap \mathcal{H}^{i.l-1}(R_3) = \emptyset$. But from lemma 6.5.1.5, $|\mathcal{H}^{i.l-1}(R_3)| = l - 1$, and because

$|\Psi^{i+1}_{R_2}| \leq |\Psi^{i+1}_K| - l$, there must be a $\psi \in \Psi^{i+1}_K \setminus \mathcal{H}^{i.l-1}(R_3)$ such that $R_2 \nvdash \psi$. Therefore there must be an $< R_4, \psi > \in \mathcal{V}$ such that $R_2 \subseteq R_4$, and $\psi \notin \mathcal{H}^{i.l-1}(R_3)$. So we have $R_2 \subseteq R_3 \cap R_4$, with $R_3 \in \mathcal{H}^{i.l-1}_{def}$, $R_4 \in < R_4, \psi > \in \mathcal{V}$ and $\psi \notin \mathcal{H}^{i.l-1}(R_3)$. So if $R_5 = R_3 \cap R_4$, then $R_5 \in \mathcal{N}$. But because $R_2 \subseteq R_3$ and $R_2 \subseteq R_4$, $R_2 \subseteq R_5$ must hold, and because $R \subset R_2$, $R \subset R_5$ must hold as well, and therefore $R \notin \max_{\subseteq} \mathcal{N}$, which contradicts the hypothesis.

For the right inclusion, i.e. $\mathcal{H}^{i.l}_{def} = \max_{\subseteq} \mathcal{N} \supseteq \max_{\subseteq} \mathcal{M}$, let $R \in \max_{\subseteq} \in \mathcal{M}$. Then $|\Psi^{i+1}_R| \leq |\Psi^{i+1}_K| - l$, and so $|\Psi^{i+1}_R| \leq (|\Psi^{i+1}_K| - l) + 1 = |\Psi^{i+1}_K| - (l-1)$ as well, such that $R \in \{Q \subseteq K \mid |\Psi^{i+1}_Q| \leq |\Psi^{i+1}_K| - (l-1)\}$. So there must be an $R_2 \in \max_{\subseteq}\{Q \subseteq K \mid |\Psi^{i+1}_Q| \leq |\Psi^{i+1}_K| - (l-1)\}$ such that $R \subseteq R_2$, and by IH, $R_2 \in \mathcal{H}^{i.l-1}_{def}$. From lemma 6.5.1.6, $\mathrm{Cn}(R_2) \cap \mathcal{H}^{i.l-1}(R_2) = \emptyset$, and from lemma 6.5.1.5, $|\mathcal{H}^{i.l-1}(R_2)| = l-1$. But $|\Psi^{i+1}_R| \leq |\Psi^{i+1}_K| - l$, so there must be a $\psi \in \Psi^{i+1}_K \setminus \mathcal{H}^{i.l-1}(R_2)$ such that $R \nvdash \psi$. Therefore there must be an R_3 such that $R \subseteq R_3$ and $< R_3, \psi > \in \mathcal{V}$. So if $R_4 = R_2 \cap R_3$, we have $R_4 \in \mathcal{N}$ and $R \subseteq R_4$. Then if $R_4 \in \mathcal{N}$, there is an $R_5 \in \max_{\subseteq} \mathcal{N} = \mathcal{H}^{i.l}_{def}$ such that $R_4 \subseteq R_5$, and $R \subseteq R_5$ holds as well. Because $R_5 \in \mathcal{H}^{i.l}_{def}$, $\mathcal{H}^{i.l}(R_5) \subseteq \Psi^{i+1}_K$ from lemma 6.5.1.4, $|\mathcal{H}^{i.l}(R_5)| = l$ from lemma 6.5.1.5, and $\mathrm{Cn}(R_5) \cap \mathcal{H}^{i.l}(R_5) = \emptyset$ from lemma 6.5.1.6. Therefore there are at least l formulas in $\Psi^{i+1}_K \setminus \mathrm{Cn}(R_5)$, or in other words $|\Psi^{i+1}_{R_5}| \leq |\Psi^{i+1}_K| - l$, such that $R_5 \in \mathcal{M}$. Now let us assume by contradiction that $R \notin \max_{\subseteq} \mathcal{N}$. Then because $R_5 \in \max_{\subseteq} \mathcal{N}$, $R \neq R_5$, and because $R \subseteq R_5$, $R \subset R_5$ must hold. But $R_5 \in \mathcal{M}$ as well, such that $R \notin \max_{\subseteq} \mathcal{M}$, contradicting the hypothesis. □

Lemma 6.5.1.8. For $0 \leq i \leq p$, if $G \in \mathcal{G}^{i,h(i)}$, $Q \subseteq G$ and $\Psi^{\geq i}_Q \neq \emptyset$, then for all $0 \leq j \leq i$, $|\Psi^j_Q| = h(j)$

Proof. By induction on i. For the base case $i = 0$, $h(0) = 0$, so if $G \in \mathcal{G}^{0,0}$, then $|\Psi^0_G| = 0$, from the definition of $\mathcal{G}$, and by monotonicity, if $Q \subseteq G$, $|\Psi^0_Q| = 0$ as well,

so for all $0 \leq j \leq 0$, $|\Psi^0_Q| = h(0)$.

For the inductive case, by IH, $|\Psi^j_Q| = h(j)$ for all $0 \leq j < i$. So we only need to show that $|\Psi^i_Q| = h(i)$. From the definition of h, because $|\Psi^j_Q| = h(j)$ for all $0 \leq j < i$ and $\Psi^{>i}_Q \neq \emptyset$, $|\Psi^i_Q| \geq h(i)$ must hold. And from the definition of $\mathcal{G}$, because $G \in \mathcal{G}^{i,h(i)}$, $|\Psi^i_G| = h(i)$. Then by monotonicity, because $Q \subseteq G$, $|\Psi^i_Q| \leq h(i)$, must hold, such that $|\Psi^i_Q| = h(i)$. □

Lemma 6.5.1.9. For $t \leq i < p$ and $0 \leq l \leq |\Psi^{i+1}_K|$, if $\mathcal{Q}^i = \max_\subseteq(\mathcal{G}^{i,h(i)})$, then $\mathcal{Q}^{i.l}_2 = \max_\subseteq(\mathcal{G}^{i+1.|\Psi^{i+1}_K|-l+1})$

Proof. The following observation is immediate from the description of function REFINE: $Q_2 \in \mathcal{Q}^{i.l}_2$ iff $Q_2 \in \max_\subseteq\{Q_1 \cap R \mid \ < Q_1, R > \in \mathcal{Q}^i \times \mathcal{H}^{i.l-1}_{def}\}$ and $\Psi^{>i+1}_{Q_2} \neq \emptyset$. So from lemma 6.5.1.1, $\mathcal{Q}^{i.l}_2 = \max \subseteq \{Q_1 \cap R \mid \ < Q_1, R > \in \mathcal{Q}^i \times \mathcal{H}^{i.l-1}_{def}$ and $\Psi^{>i+1}_{Q_1 \cap R} \neq \emptyset\}$. As a notational shortcut, let $\mathcal{P} = \{Q \cap R \mid \ < Q, R > \in \mathcal{Q}^i \times \mathcal{H}^{i.l-1}_{def}$ and $\Psi^{>i+1}_{Q \cap R} \neq \emptyset\}$, such that $\mathcal{Q}^{i.l}_2 = \max_\subseteq \mathcal{P}$.

Let us start with the left inclusion, i.e. if $\mathcal{Q}^i = \max_\subseteq(\mathcal{G}^{i,h(i)})$, then $\mathcal{Q}^{i.l}_2 \subseteq \max_\subseteq(\mathcal{G}^{i+1.|\Psi^{i+1}_K|-l+1})$. Let $Q_2 \in \mathcal{Q}^{i.l}_2$. Then because $\mathcal{Q}^{i.l}_2 \subseteq \mathcal{P}$, $Q_2 \in \mathcal{P}$, an therefore $\Psi^{>i+1}_{Q_2} \neq \emptyset$, so $\Psi^{>i}_{Q_2} \neq \emptyset$. Because $Q_2 \in \mathcal{P}$, we also have $Q_2 = Q \cap R$ for some $< Q, R > \in \mathcal{Q}^i \times \mathcal{H}^{i.l-1}_{def}$. From the hypothesis, $\mathcal{Q}^i = \max_\subseteq(\mathcal{G}^{i,h(i)})$, so $Q \in \mathcal{G}^{i,h(i)}$. And because $Q_2 \subseteq Q$ and $\Psi^{>i}_{Q_2} \neq \emptyset$, from lemma 6.5.1.8, for all $0 \leq j \leq i$, $\Psi^j_{Q_2} = h(j)$. In addition, from lemma 6.5.1.7, $R \in \max_\subseteq\{Q' \subseteq K \mid |\Psi^{i+1}_{Q'}| \leq |\Psi^{i+1}_K| - (l-1)\}$, so $|\Psi^{i+1}_R| \leq |\Psi^{i+1}_K| - (l-1)$, and by monotonicity, $|\Psi^{i+1}_{Q_2}| \leq |\Psi^{i+1}_K| - (l-1)$ as well. So we have $\Psi^j_{Q_2} = h(j)$ for all $0 \leq j \leq i$, $|\Psi^{i+1}_{Q_2}| \leq |\Psi^{i+1}_K| - (l-1)$, and $\Psi^{>i+1}_{Q_2} \neq \emptyset$, which from the definition of $\mathcal{G}$ imply that $Q_2 \in \mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1}$.

Now let us assume by contradiction that $Q_2 \notin \max_\subseteq(\mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1})$. Then there must be a $G \in \mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1}$ such that $Q_2 \subset G$. Because $G \in \mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1}$, for all $0 \leq j \leq i$, $\Psi^j_G = h(j)$. So there must be a maximal subset Q_3 of K such that

$G \subseteq Q_3$ and for all $0 \leq j \leq i$, $\Psi^j_{Q_3} = h(j)$. Then because $G \in \mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1}$, from the definition of $\mathcal{G}$, $\Psi^{>i+1}_G \neq \emptyset$, so by monotonicity, $\Psi^{>i+1}_{Q_3} \neq \emptyset$ as well, and therefore $\Psi^{>i}_{Q_3} \neq \emptyset$. So from the definition of $\mathcal{G}$, $Q_3 \in \mathcal{G}^{i,h(i)}$, such that there must be a $Q_4 \in \max_{\subseteq} \mathcal{G}^{i,h(i)}$ with $Q_3 \subseteq Q_4$, and from the hypothesis, $Q_4 \in \mathcal{Q}^i$. In addition, because $G \in \mathcal{G}^{i+1,|\Psi^{i+1}_K|-(l-1)}$, $|\Psi^{i+1}_G| \leq |\Psi^{i+1}_K| - (l-1)$ must hold, and so there is a maximal $R_2 \subseteq K$ such that $|\Psi^{i+1}_{R_2}| \leq |\Psi^{i+1}_K| - (l-1)$ and $G \subseteq R_2$, and from lemma 6.5.1.7, $R_2 \in \mathcal{H}^{i.l-1}_{def}$. So $G \subseteq Q_4 \cap R_2$. Now take $Q_5 = Q_4 \cap R_2$. Then $G \subseteq Q_5$, and by monotonicity, because $\Psi^{>i+1}_G \neq \emptyset$, $\Psi^{>i+1}_{Q_5} \neq \emptyset$ as well. Therefore $Q_5 \in \mathcal{P}$. But because $Q_2 \subset G$ and $G \subseteq Q_5$, $Q_2 \subset Q_5$ must hold, such that $Q_2 \notin \max_{\subseteq} \mathcal{P} = \mathcal{Q}^{i.l-1}_2$, contradicting the hypothesis.

Now let us continue with the right inclusion, i.e. if $\mathcal{Q}^i = \max_{\subseteq}(\mathcal{G}^{i,h(i)})$, then $\mathcal{Q}^{i.l}_2 = \max_{\subseteq} \mathcal{P} \supseteq \max_{\subseteq}(\mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1})$. Let $G \in \max_{\subseteq} \mathcal{G}^{i+1,|\Psi^{i+1}_K|-l+1}$. Then from the definition of $\mathcal{G}$, $|\Psi^{i+1}_G| \leq |\Psi^{i+1}_K| - l + 1 = |\Psi^{i+1}_K| - (l-1)$, so from lemma 6.5.1.7, there must be an $R \in \mathcal{H}^{i.l-1}_{def}$ such that $G \subseteq R$. From the definition of $\mathcal{G}$ still, $\Psi^j_G = h(j)$ for all $0 \leq j \leq i$, so there must be a maximal subset Q of K such that $G \subseteq Q$ and $\Psi^j_Q = h(j)$ for all $0 \leq j \leq i$. Finally, $\Psi^{>i+1}_G \neq \emptyset$, so by monotonicity $\Psi^{>i+1}_Q \neq \emptyset$, and therefore $\Psi^{>i}_Q \neq \emptyset$. So $Q \in \mathcal{G}^{i,h(i)}$. Therefore there must be a $Q_2 \in \max_{\subseteq} \mathcal{G}^{i,h(i)}$ such that $Q \subseteq Q_2$, and from the hypothesis, $Q_2 \in \mathcal{Q}^i$. Now let $Q_3 = Q_2 \cap R$. Then $G \subseteq Q_3$, and because $\Psi^{>i+1}_G \neq \emptyset$, by monotonicity, $\Psi^{>i+1}_{Q_3} \neq \emptyset$ as well, such that $Q_3 \in \mathcal{P}$. Therefore there must be a $Q_4 \in \max_{\subseteq} \mathcal{P}$ such that $Q_3 \subseteq Q_4$, and by the transitivity of $\subseteq$, $G \subseteq Q_4$. Because $Q_4 \in \mathcal{P}$, $\Psi^{>i+1}_{Q_4} \neq \emptyset$, and so $\Psi^{>i}_{Q_4}$ must hold as well. In addition, because $Q_4 \in \mathcal{P}$, there is a $< Q_5, R_2 > \in \mathcal{Q}^i \times \mathcal{H}^{i.l-1}_{def}$ such that $Q_4 = Q_5 \cap R_2$, and from the hypothesis, $Q_5 \in \mathcal{G}^{i,h(i)}$. So we have $Q_5 \in \mathcal{G}^{i,h(i)}$, $\Psi^{>i}_{Q_4}$ and $Q_4 \subseteq Q_5$, such that from lemma 6.5.1.8, $\Psi^j_{Q_4} = h(j)$ for all $0 \leq j \leq i$. And from lemma 6.5.1.7, because $R_2 \in \mathcal{H}^{i.l-1}_{def}$, we have $|\Psi^{i+1}_{R_2}| \leq |\Psi^{i+1}_K| - (l-1) = |\Psi^{i+1}_K| - l + 1$, so by monotonicity $|\Psi^{i+1}_{Q_4}| \leq |\Psi^{i+1}_K| - l + 1$ as well. So from the definition of $\mathcal{G}$,

$Q_4 \in \mathcal{G}^{i+1,|\Psi_K^{i+1}|-l+1}$.

Now let us assume by contradiction that $G \notin \max_\subseteq \mathcal{P}$. Because $Q_4 \in \max_\subseteq \mathcal{P}$, $G \neq Q_4$, and because $G \subseteq Q_4$, $G \subset Q_4$ must hold. But $Q_4 \in \mathcal{G}^{i+1,|\Psi_K^{i+1}|-l+1}$, such that $G \notin \max_\subseteq \mathcal{G}^{i+1,|\Psi_K^{i+1}|-l+1}$, contradicting the hypothesis. □

6.5.1.6.2 Main proposition

Proposition. For $t \leq i \leq p$, $\mathcal{Q}^i = \max_\subseteq \mathcal{G}^{i,h(i)}$

Proof. By induction on i. For the base case $i = t$, from line 1 of algorithm 3, $\mathcal{Q}^t = \max_\subseteq \mathcal{B}^t$. So we need to show that $\max_\subseteq \mathcal{B}^t = \max_\subseteq \mathcal{G}^{t,h(t)}$. From the definition of $\mathcal{G}$, $G \in \mathcal{G}^{t,h(t)}$ iff $G \subseteq K$ and the two following hold:

- for all $0 \leq j \leq t$, $|\Psi_G^j| = h(j)$
- $\Psi_G^{>t} \neq \emptyset$

From the definition of t, $\mathcal{B}^t \neq \emptyset$, and so there is a $B \in \mathcal{B}^t$ such that $\Psi_B^{>t} \neq \emptyset$ and $\Psi_B^{\leq t} = \emptyset$, and therefore for each $0 \leq j \leq t$, $h(j) = 0$. Additionally, for any $B \in \mathcal{B}^t$, $\Psi_B^{>t} \neq \emptyset$ and $\Psi_B^{\leq t} = \emptyset$, so $B \in \mathcal{G}^{t,h(t)}$. Similarly, from the definition of $\mathcal{G}$, for any $G \in \mathcal{G}^{t,h(t)}$, $\Psi_G^{>t} \neq \emptyset$, and for all $1 \leq j \leq t$, $|\Psi_G^j| = h(j) = 0$, so $\Psi_G^{\leq t} = \emptyset$, and therefore $G \in \mathcal{B}^t$. So $\mathcal{B}^t = \mathcal{G}^{t,h(t)}$, and as a consequence, $\max_\subseteq \mathcal{B}^t = \max_\subseteq \mathcal{G}^{t,h(t)}$.

For the inductive case, let $\mathcal{Q}^i = \max_\subseteq \mathcal{G}^{i,h(i)}$ hold by IH. Then we need to show that $\mathcal{Q}^{i+1} = \max_\subseteq \mathcal{G}^{i+1,h(i+1)}$ if $i+1 \leq p$. This choice is made (instead of i and $i-1$) in order to follow the notation used in the description of function Refine. From lemma 6.5.1.9, because $\mathcal{Q}^i = \max_\subseteq \mathcal{G}^{i,h(i)}$, $\mathcal{Q}_2^{i.l} = \max_\subseteq(\mathcal{G}^{i+1,|\Psi_K^{i+1}|-l+1})$. As a consequence, $\mathcal{Q}_2^{i.l} = \emptyset$ iff $\mathcal{G}^{i+1,|\Psi_K^{i+1}|-l+1} = \emptyset$, and from the description of function Refine line 6, the function must return when $\mathcal{Q}_2 = \emptyset$. Then because $i+1 \leq p$, h is defined for $i+1$, and from the definition of h, $\mathcal{G}^{i+1,|\Psi_K^{i+1}|-l+1} = \emptyset$ iff $|\Psi_K^{i+1}|-l+1 < h(i+1)$, i.e iff $l > |\Psi_K^{i+1}|-$

$h(i+1)+1$. So the function REFINE must return exactly after $|\Psi_K^{i+1}| - h(i+1)+2$ iterations over its main loop, i.e. when l reaches $|\Psi_K^{i+1}| - h(i+1)+2$. At this stage, $\mathcal{Q}_2^{i.l} = \emptyset$, and so from line 7 of the description of function REFINE, the value returned by the function is $\mathcal{O}_2^{i.l} = \mathcal{Q}_2^{i.l-1} = \mathcal{Q}_2^{i.|\Psi_K^{i+1}|-h(i+1)+1}$. Let $v = |\Psi_K^{i+1}| - h(i+1)+1$, such that the value returned by function REFINE is $\mathcal{Q}_2^{i.v}$. Then from lemma 6.5.1.9, and the fact that $\mathcal{Q}^i = \max_{\subseteq} \mathcal{G}^{i,h(i)}$, $\mathcal{Q}_2^{i.v} = \max_{\subseteq} \mathcal{G}^{i+1.|\Psi_K^{i+1}|-v+1}$. Replacing v by $|\Psi_K^{i+1}| - h(i+1)+1$, this yields $\max_{\subseteq} \mathcal{G}^{i+1.|\Psi_K^{i+1}|-(|\Psi_K^{i+1}|-h(i+1)+1)+1} = \max_{\subseteq} \mathcal{G}^{i+1.h(i+1)}$. So in algorithm 3 line 4, the function REFINE returns $\max_{\subseteq} \mathcal{G}^{i+1.h(i+1)}$, such that immediately after incrementing i line 5, $\mathcal{Q}^i = \max_{\subseteq} \mathcal{G}^{i,h(i)}$. □

6.5.1.7 Proposition 6.3.2.10

Proposition. For $p \leq i < m$, $\mathcal{Q}^{i+1} = \mathcal{Q}^i$

Proof. By induction on i. For the base case $i = p$, when variable i reaches p, because $p < m$, the main loop of algorithm 3 is executed at least once again, such that from proposition 6.3.2.9, immediately before the call to the function REFINE, line 4, the value of variable $\mathcal{Q}$ is $\mathcal{Q}^p = \mathcal{G}^{p,h(p)}$. Then in function REFINE, variable Q_2 line 11 must be a subset of some $G \in \mathcal{Q}^p = \mathcal{G}^{p,h(p)}$. Because $G \in \mathcal{G}^{p,h(p)}$, for all $0 \leq j \leq p$, $|\Psi_G^j| = h(j)$, and because $Q_2 \subseteq G$, by monotonicity, $|\Psi_{Q_2}^j| \leq h(j)$. Now let us assume that $\Psi_{Q_2}^j < h(j)$ for some $0 \leq j \leq p$. Take the smallest j verifying this property. From the definition of h, there is no $Q \subseteq K$ such that $\Psi_Q^j = h(k)$ for all $0 \leq k < j$, and $\Psi_Q^j < h(j)$ and $\Psi_Q^{>j} \neq \emptyset$. So $\Psi_{Q_2}^{>j} = \emptyset$ must hold, and because $j \leq p$, $\Psi_{Q_2}^{>p+1} = \emptyset$ must hold as well. And if $\Psi_{Q_2}^j = h(j)$ for all $0 \leq j \leq p$. from the definition of p, $\Psi_{Q_2}^{>p+1} = \emptyset$ as well. Then because, $\Psi_{Q_2}^{>p+1} = \emptyset$, the condition line 18 is never met, Q_2 is not retained in $\mathcal{Q}_2$, and as a consequence, line 6 $\mathcal{Q}_\in = \{\emptyset\}$, such that the variable $\mathcal{O}_2$ is not updated, and the function REFINE returns its input, i.e. $\mathcal{Q}^p$. So back to

the function call in algorithm 3 line 4, $\mathcal{Q}$ remains unchanged, such that $\mathcal{Q}^{p+1} = \mathcal{Q}^p$.

For the inductive case, the proof is almost identical. By IH, $\mathcal{Q}^i = \mathcal{Q}^{i-1} = \mathcal{Q}^p$, so variable Q_2 must be a subset of some $G \in \mathcal{G}^{p,h(p)}$ as well. Then for the same reason as previously, $\Psi_{Q_2}^{>p+1} = \emptyset$ must hold, and because $p < i$, $\Psi_{Q_2}^{>i+1} = \emptyset$ holds as well, such that line 18, Q_2 is not retained in $\mathcal{Q}_2$, and the function REFINE returns its input, i.e. $\mathcal{Q}^i$, and therefore $\mathcal{Q}^{i+1} = \mathcal{Q}^i$. □

6.5.1.8 Proposition 6.3.2.11

Proposition. $\max_{\preceq_{lex_K}} 2^K = \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$

6.5.1.8.1 Lemmas

Lemma 6.5.1.10. for $0 \leq i \leq p$, $\mathcal{G}^{i,h(i)} \neq \emptyset$

Proof. Because $i \leq p$, $h(i)$ is defined, and from the definition of h, there is a $Q \subseteq K$ verifying $|\Psi_Q^{>i}| \neq \emptyset$, $|\Psi_Q^j| = h(j)$ for all $1 \leq j < i$, and $|\Psi_Q^i| = h(i)$, and therefore immediately from the definition of $\mathcal{G}$, $\mathcal{G}^{i,h(i)} \neq \emptyset$. □

For readability, Proposition 6.3.2.3 is reproduced here.

Proposition. $Q \prec_{lex_K} Q_2$ iff either:

(1) for all $1 \leq i \leq m$, $|\Psi_{Q_1}^i| = |\Psi_{Q_2}^i|$, and $Q_1 \subset Q_2$, or

(2) there is a $1 \leq i \leq m$ such that for all $1 \leq j < i$ $|\Psi_{Q_1}^j| = |\Psi_{Q_2}^j|$, and one of the three following conditions holds:

(2.1) $\Psi_{Q_1}^{>i} = \emptyset$ and $\Psi_{Q_2}^{>i} \neq \emptyset$

(2.2) $\Psi_{Q_2}^{>i} \neq \emptyset$, and $|\Psi_{Q_2}^i| < |\Psi_{Q_1}^i|$

(2.3) $\Psi^{>i}_{Q_1} = \emptyset$, $\Psi^{>i}_{Q_2} = \emptyset$, and $|\Psi^i_{Q_2}| > |\Psi^i_{Q_1}|$

Lemma 6.5.1.11. If $Q \in \max_{\preceq_{lex_K}} 2^K$, then $Q \in \max_{\subseteq} \mathcal{G}^{p,h(p)}$

Proof. By contraposition. Let $Q_1 \notin \max_{\subseteq} \mathcal{G}^{p,h(p)}$. Then we need to show that $Q_1 \notin \max_{\preceq_{lex_K}} 2^K$, i.e. that there is a Q_2 such that $Q_1 \prec_{lex_K} Q_2$. If $Q_1 \notin \max_{\subseteq} \mathcal{G}^{p,h(p)}$, then either $Q_1 \notin \mathcal{G}^{p,h(p)}$, or $Q_1 \in \mathcal{G}^{p,h(p)}$ but $Q_1 \notin \max_{\subseteq} \mathcal{G}^{p,h(p)}$.

Let us start with the first possibility, i.e. $Q_1 \notin \mathcal{G}^{p,h(p)}$. From the definition of $\mathcal{G}$, one of the two following must hold:

- (a) there is a $0 \leq i \leq p$, such that $|\Psi^i_{Q_1}| \neq h(i)$
- (b) $\Psi^{>p}_{Q_1} = \emptyset$

From lemma 6.5.1.10, $\mathcal{G}^{p,h(p)} \neq \emptyset$. Take any $Q_2 \in \mathcal{G}^{p,h(p)}$. Let us consider possibility (a) first, i.e. the case where there is a $0 \leq i \leq p$, such that $|\Psi^i_{Q_1}| \neq h(i)$, and take the smallest i verifying this property. From the definition of $\mathcal{G}$, $\Psi^{>p}_{Q_2} \neq \emptyset$, so because $i \leq p$, $\Psi^{>i}_{Q_2} \neq \emptyset$ as well. Additionally, for all $0 \leq j < i$, $\Psi^j_{Q_1} = \Psi^j_{Q_2} = h(j)$. So if $\Psi^{>i}_{Q_1} = \emptyset$, from proposition 6.3.2.3 property (2.1), $Q_1 \prec_{lex_K} Q_2$ must hold. If $\Psi^{>i}_{Q_1} \neq \emptyset$, from the definition of h, $|\Psi^i_{Q_1}| \geq h(i)$, and because $\Psi^i_{Q_1} \neq h(i)$, $|\Psi^i_{Q_1}| > h(i) = |\Psi^i_{Q_2}|$ must hold, so from proposition 6.3.2.3 property (2.2), $Q_1 \prec_{lex_K} Q_2$ holds as well.

Now for possibility (b), let us assume that $\Psi^{>p}_{Q_1} = \emptyset$, but for all $0 \leq i \leq p$, $|\Psi^i_{Q_1}| = |\Psi^i_{Q_2}| = h(i)$. In this case too, because $\Psi^{>p}_{Q_2} \neq \emptyset$, from proposition 6.3.2.3 property (2.1), $Q_1 \prec_{lex_K} Q_2$ must hold.

So if $Q_1 \notin \mathcal{G}^{p,h(p)}$, then $Q_1 \notin \max_{\prec_{lex_K}} 2^K$. Now let us assume that $Q_1 \in \mathcal{G}^{p,h(p)}$, but $Q_1 \notin \max_{\subseteq} \mathcal{G}^{p,h(p)}$. Then there is a $Q_2 \in \mathcal{G}^{p,h(p)}$ such that $Q_1 \subset Q_2$. From the definition of $\mathcal{G}$, for all $0 \leq i \leq p$, $|\Psi^i_{Q_1}| = |\Psi^i_{Q_2}| = h(i)$. And from the definition of p, $\Psi^{>p+1}_{Q_1} = \Psi^{>p+1}_{Q_2} = \emptyset$. So for all $0 \leq i \leq m$, $|\Psi^i_{Q_1}| = |\Psi^i_{Q_2}|$, and because $Q_1 \subset Q_2$, from proposition 6.3.2.3 property (1), $Q_1 \prec_{lex_K} Q_2$. □

6.5.1.8.2 Main proposition

Proposition. $\max_{\preceq_{lex_K}} 2^K = \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$

Proof. For the left inclusion, we need to show that $\max_{\preceq_{lex_K}} 2^K \subseteq \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$. Let $Q \in \max_{\preceq_{lex_K}} 2^K$. Then from lemma 6.5.1.11, $Q \in \max_{\subseteq} \mathcal{G}^{p,h(p)}$. Now let us assume by contradiction that $Q \notin \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$. Because $Q \in \max_{\subseteq} \mathcal{G}^{p,h(p)}$, there must be a $Q' \in \max_{\subseteq} \mathcal{G}^{p,h(p)}$ such that $Q \prec_{lex_K} Q'$. Now from the definition of $\mathcal{G}$, $\mathcal{G}^{p,h(p)} \subseteq 2^K$. So $\max_{\subseteq} \mathcal{G}^{p,h(p)} \subseteq 2^K$ as well, such that $Q' \in 2^K$. Then because $Q \in \max_{\preceq_{lex_K}} 2^K$, $Q' \preceq_{lex_K} Q$ must hold, which contradicts $Q \prec_{lex_K} Q'$.

For the right inclusion, we need to show that $\max_{\preceq_{lex_K}} 2^K \supseteq \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$. Let $Q \in \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$. From the definition of $\mathcal{G}$, $\mathcal{G}^{p,h(p)} \subseteq 2^K$. So $\max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)} \subseteq 2^K$ as well, such that $Q \in 2^K$. Now let us assume by contradiction that $Q \notin \max_{\preceq_{lex_K}} 2^K$. Because $Q \in 2^K$, there must be a $Q' \in 2^K$ such that $Q \prec_{lex_K} Q'$, and a $Q'' \in \max_{\preceq_{lex_K}} 2^K$ such that $Q \prec_{lex_K} Q' \preceq_{lex_K} Q''$. But from lemma 6.5.1.11, $Q'' \in \max_{\subseteq} \mathcal{G}^{p,h(p)}$. So because $Q \in \max_{\preceq_{lex_K}} \max_{\subseteq} \mathcal{G}^{p,h(p)}$, $Q'' \preceq_{lex_K} Q$ must hold, which contradicts $Q \prec_{lex_K} Q''$. □

6.5.2 Complexity of algorithm 3

6.5.2.1 Proposition 6.3.2.13

Proposition. $g(n) = \Omega(c^n)$, with $c > 1$

First, one may observe that in the worst case, the number of base remainders for a given consequence ψ can be the cardinality of some maximal subset $\mathcal{M}$ of 2^K such that for all $M_1, M_2 \in \mathcal{M}$, $M_1 \not\subseteq M_2$. If $|K|$ is odd, $\mathcal{M}$ is the family of all subsets of K either of size $\lceil \frac{|K|}{2} \rceil$ or of size $\lfloor \frac{|K|}{2} \rfloor$. For instance, if $K = \{\phi_1, \phi_2, \phi_3\}$, then it may be the case for some $\psi \in \mathrm{Cn}(K)$ that $\mathcal{R}_{\subseteq}(\psi, K) = \{\{\psi_1, \psi_2\}, \{\psi_2, \psi_3\}, \{\psi_1, \psi_3\}\}$. If $|K|$

is even, $\mathcal{M}$ is the family of all subsets of K of size $\frac{|K|}{2}$. But in both cases (whether K is odd or even), $|M| = \binom{|K|}{\lceil \frac{|K|}{2} \rceil}$. Therefore whichever the DL under consideration is, $g(n) = \Omega(\binom{n}{\lceil \frac{n}{2} \rceil})$. Or to simplify notation, if $q : \mathbb{N}^+ \mapsto \mathbb{N}$ is defined by $q(n) = \binom{n}{\lceil \frac{n}{2} \rceil}$, then $g(n) = \Omega(q(n))$.

To simplify notation still, let $n^{(\underline{k})} = n(n-1)\dots(n-(k-1))$. For instance, $6^{(\underline{3})} = 6 \times 5 \times 4$.

Let us assume that $n \geq 2$, and let us start with the case where n is even. Then immediately from the definition of the binomial coefficient, if $q(n) = \binom{n}{\lceil \frac{n}{2} \rceil}$, the following must hold:

$$q(n) = \frac{n^{(\underline{\frac{n}{2}})}}{(\frac{n}{2})!}$$

And because $\frac{n}{2} = \lceil \frac{n-1}{2} \rceil$ when n is even, we have:

$$q(n) = \frac{n^{(\underline{\lceil \frac{n-1}{2} \rceil})}}{(\lceil \frac{n-1}{2} \rceil)!}$$

$$q(n) = \frac{n}{\lceil \frac{n}{2} \rceil} \cdot \frac{(n-1)^{(\underline{\lceil \frac{n-1}{2} \rceil})}}{(\lceil \frac{n-1}{2} \rceil)!}$$

$$q(n) = \frac{n}{\lceil \frac{n}{2} \rceil} \cdot q(n-1)$$

For instance, if $n = 6$, then $q(n-1) = \binom{5}{3} = \frac{5^{\underline{3}}}{3!} = \frac{5 \times 4 \times 3}{3!}$,
and $q(n) = \binom{6}{3} = \frac{6^{\underline{3}}}{3!} = \frac{6 \times 5 \times 4}{3!} = \frac{6 \times 5 \times 4 \times 3}{3! \times 3} = \frac{6}{3} \cdot \binom{5}{3}$
If n is odd instead, then:

$$q(n) = \frac{n^{\underline{\lceil \frac{n}{2} \rceil}}}{\lceil \frac{n}{2} \rceil!}$$

And because $\lceil \frac{n}{2} \rceil = \frac{n-1}{2} + 1$ when n is odd, we have:

$$q(n) = \frac{n^{(\underline{\frac{n-1}{2}+1})}}{(\frac{n-1}{2}+1)!}$$

$$q(n) = \frac{n}{\lceil \frac{n}{2} \rceil} \cdot \frac{(n-1)^{(\underline{\frac{n-1}{2}})}}{(\frac{n-1}{2})!}$$

$$q(n) = \frac{n}{\lceil \frac{n}{2} \rceil} \cdot q(n-1)$$

For instance, if $n = 5$, then $q(n-1) = \binom{4}{2} = \frac{4^2}{2!} = \frac{4\times 3}{2!}$,
and $q(n) = \binom{5}{3} = \frac{5^3}{3!} = \frac{5\times 4\times 3}{3\times 2!} = \frac{5}{3} \cdot \binom{4}{2}$
So in both cases (n odd and even), the following recursion holds:

$$q(n) = \frac{n}{\lceil \frac{n}{2} \rceil} \cdot q(n-1)$$

Let $c_n = \frac{n}{\lceil \frac{n}{2} \rceil}$. Then for $n > 1$, we have $q(n) = q(1) \times c_2, .., \times c_n = 1 \times c_2, .., \times c_n$. If n is even, then $c_n = 2$, whereas the minimal value for c_n when n is odd is $c_3 = \frac{3}{2} < 2$. Therefore for $n > 1$, we have $q(n) > (\frac{3}{2})^n$. So if $c = c_3 = \frac{3}{2}$, we have $q(n) > c^n$, and because $g(n) = \Omega(q(n))$, $g(n) = \Omega(c^n)$ must hold as well. Finally, because $c = \frac{3}{2} > 1$, $g(n)$ is at least exponential in n the worst case.

6.5.2.2 Proposition 6.3.2.14

Proposition. $f(n) = O(|\Psi_K| \cdot g(n))$

6.5.2.2.1 Lemmas

Lemma 6.5.2.1. Let $K \nvdash \bot$, and $\Psi \subseteq \mathrm{Cn}(K)$ such that for all $\psi \in \Psi$, $\mathrm{Cn}(\psi) \neq \mathrm{Cn}(\emptyset)$. If $\Psi = \Psi_1 \cup \Psi_2$, then $\mathcal{R}^{\vee}_{\subseteq}(\Psi, K) = \max_{\subseteq}\{R_1 \cap R_2 \mid < R_1, R_2 > \in \mathcal{R}^{\vee}_{\subseteq}(\Psi_1, K) \times \mathcal{R}^{\vee}_{\subseteq}(\Psi_2, K)\}$

Proof. Let $\mathcal{P} = \{R_1 \cap R_2 \mid < R_1, R_2 > \in \mathcal{R}^{\vee}_{\subseteq}(\Psi_1, K) \times \mathcal{R}^{\vee}_{\subseteq}(\Psi_2, K)\}$,

Let us start with the left inclusion, i.e. $\mathcal{R}^{\vee}_{\subseteq}(\Psi, K) \subseteq \max_{\subseteq} \mathcal{P}$, and let $R \in \mathcal{R}^{\vee}_{\subseteq}(\Psi, K)$. Then from the definition of $\mathcal{R}_{\subseteq}$, $\mathrm{Cn}(R) \cap \Psi = \emptyset$, and because $\Psi_1 \subseteq \Psi$ $\mathrm{Cn}(R) \cap \Psi_1 = \emptyset$ as well. So there must be an $R_1 \in \mathcal{R}^{\vee}_{\subseteq}(\Psi_1, K)$ such that $R \subseteq R_1$. Similarly, because $\Psi_2 \subseteq \Psi$, there must be an $R_2 \in \mathcal{R}^{\vee}_{\subseteq}(\Psi_1, K)$ such that $R \subseteq R_2$. Take $R_3 = R_1 \cap R_2$. Then $R_3 \in \mathcal{P}$. So there must be an $R_4 \in \max_{\subseteq} \mathcal{P}$ such that

$R_3 \subseteq R_4$. And because $R_4 \in \mathcal{P}$, it must verify $\mathrm{Cn}(R_4) \cap \Psi_1 = \emptyset$ and $\mathrm{Cn}(R_4) \cap \Psi_2 = \emptyset$, therefore $\mathrm{Cn}(R_4) \cap \Psi = \emptyset$. By the transitivity of $\subseteq$, $R \subseteq R_4$, so if $R \neq R_4$, $R \subset R_4$ must hold. But because $\mathrm{Cn}(R_4) \cap \Psi$, this would contradict $R \in \mathcal{R}_{\subseteq}^{\vee}(\Psi, K)$, so $R = R_4$ must hold, an therefore $R \in \max_{\subseteq} \mathcal{P}$.

For the right inclusion, i.e. $\mathcal{R}_{\subseteq}^{\vee}(\Psi, K) \supseteq \max_{\subseteq} \mathcal{P}$, let $R \in \max_{\subseteq} \mathcal{P}$. Then $\mathrm{Cn}(R) \cap \Psi_1 = \emptyset$ and $\mathrm{Cn}(R) \cap \Psi_2 = \emptyset$, so $\mathrm{Cn}(R) \cap \Psi = \emptyset$. Now let us assume by contradiction that $R \notin \mathcal{R}_{\subseteq}^{\vee}(\Psi, K)$. Because $\mathrm{Cn}(R) \cap \Psi = \emptyset$, the only remaining possibility is that there is an $R \subset R_2$ such that $\mathrm{Cn}(R_2) \cap \Psi = \emptyset$. Then there must be an $R_3 \in \mathcal{R}_{\subseteq}^{\vee}(\Psi, K)$ such that $R_2 \subseteq R_3$ and because $\mathrm{Cn}(R_3) \cap \Psi = \emptyset$, $R_3 \in \mathcal{P}$. But because $R \subset R_2$ and $R_2 \subseteq R_3$, $R \subset R_3$ must hold, such that $R \notin \max_{\subseteq} \mathcal{P}$, contradicting the hypothesis. □

6.5.2.2.2 Main proposition

Proposition. $f(n) = O(|\Psi_K| \cdot g(n))$

As a reminder, f is the cumulated cost of the execution of algorithms 2 and 3. and g is the cost of computing $\mathcal{R}_{\subseteq}(\psi, K)$ for some $\psi \in \Psi_K$, both being expressed as functions of $|K|$.

From proposition 6.3.2.13, $g(n) = \Omega(c^n)$, with $c > 1$.

In addition, the two following properties, which were shown to hold in Section 6.5.2.1, will be useful. Let $q : \mathbb{N}^+ \mapsto \mathbb{N}$ be defined by $q(n) = \binom{n}{\lceil \frac{n}{2} \rceil}$.

- If Γ is a finite set, and if $\mathcal{M} \subseteq 2^{\Gamma} \setminus \{\emptyset\}$ is such that for all $M_1, M_2 \in \mathcal{M}$, $M_1 \not\subseteq M_2$, then $|\mathcal{M}| \leq q(|\Gamma|)$.
- $g(n) = \Omega(q(n))$

Let us start with phase 1, i.e. algorithm 2, and let us assume temporarily that $\mathcal{R}_{\subseteq}^{\vee}(\Psi_K^i, K)$ is known for each $1 \leq i \leq m$, a well as $\mathcal{R}_{\subseteq}^{\vee}(\Psi_K^{>i}, K)$ for each $1 \leq i < m$.

A first remark is that at any state of the execution, due to lines 10 to 12 of algorithm 2 and the initialization of $\mathcal{Q}$, for all $Q_1, Q_2 \in \mathcal{Q}$, if $Q_1 \neq Q_2$, then $Q_1 \not\subseteq Q_2$. So because $\mathcal{Q} \subseteq 2^K$, from the observation made in Section 6.5.2.1, $|\mathcal{Q}| \leq q(|K|)$ must hold. This property also holds for $\mathcal{O}$, which at any state is either $\emptyset$ or a copy of a previous state of $\mathcal{Q}$, from line 5, so $|\mathcal{O}| \leq q(|K|)$. Finally, from the definition of $\mathcal{R}_{\subseteq}$, all $R \in \mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)$ are maximal wrt set inclusion, such that for all $R_1, R_2 \in \mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)$, if $R_1 \neq R_2$, then $R_1 \not\subseteq R_2$, and therefore $|\mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)| \leq q(|K|)$ holds as well. So line 9, because each Q_2 is the intersection of some $< Q_1, R > \in \mathcal{O} \times \mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)$, the number of generated intersections is $|\mathcal{O}| \times |\mathcal{R}^{\vee}_{\subseteq}(\Psi^{i+1}_K, K)|$, which is (loosely) bounded by $q(|K|)^2$. Then lines 10 an 11, Q_2 is compared wrt set inclusion to each $Q \in \mathcal{Q}$ (comparing once Q_2 to each $Q \in \mathcal{Q}$ is enough for the execution of both lines 10 and 11), so the number of comparisons performed at each execution of lines 10 and 11 is bounded by $q(|K|)^2 \times |\mathcal{Q}| < q(|K|)^3$. Then line 16 verifies for each $Q_2 \in \mathcal{Q}$ whether $\Psi^{>i+1}_{Q_2} = \emptyset$. For a given Q_2 and a given i, although it is arguably very inefficient, this is equivalent to verifying whether there is an $R \in \mathcal{R}^{\vee}_{\subseteq}(\Psi^{>i+1}_K, K)$ such that $Q_2 \subseteq R$. So if $\mathcal{R}^{\vee}_{\subseteq}(\Psi^{>i+1}_K, K)$ is known (which, again, is temporarily assumed), the number of pairwise comparisons line 16 between an element of $\mathcal{Q}$ and an element of $\mathcal{R}^{\vee}_{\subseteq}(\Psi^{>i+1}_K, K)$ is once again bounded by $q(|K|)^2$. To sum up, during each iteration of the main loop of algorithm 2, less than $q(|K|)^2$ pairwise intersections of subsets of K are computed line 9, less than $q(|K|)^3$ pairwise comparisons of subsets of K wrt set inclusion are preformed lines 10 and 11, and less than $q(|K|)^2$ pairwise comparisons of subsets of K wrt set inclusion are preformed line 16. Finally, the number of iterations over the main loop of algorithm 2 is bounded by m.

Let $f_1(n)$ express the cost of algorithm 2 as a function of $|K|$, assuming still temporarily that $\mathcal{R}^{\vee}_{\subseteq}(\Psi^{i}_K, K)$ is known for each $1 \leq i \leq m$, a well as $\mathcal{R}^{\vee}_{\subseteq}(\Psi^{>i}_K, K)$ for each $1 \leq i < m$. Comparing two sets of cardinality $\leq n$ wrt to set inclusion or

computing their intersection is in $O(n \log n)$. Therefore from the above observations, $f_1(n) = O(m(q(n)^2 n \log n + q(n)^3 n \log n + q(n)^2 n \log n)) = O(mn \log n(2q(n)^2 + q(n)^3)) = O(mq(n)^3)$.

Now let us focus on the computation of all $\mathcal{R}^{\vee}_{\subseteq}(\Psi^i_K, K)$ for $1 \leq i \leq m$ (which were temporarily assumed to be known). From lemma 6.5.2.2.1, the following (poorly optimized) strategy can in theory be applied:

- Compute $\mathcal{R}_{\subseteq}(\psi, K)$ for each $\psi \in \Psi^i_K$. The cumulated cost is bounded by $|\Psi^i_K| \cdot g(|K|)$.
- If $\Psi^i_K = \{\psi_1, .., \psi_l\}$, iteratively compute $\mathcal{R}^{\vee}_{\subseteq}(\{\psi_1, .., \psi_k\}, K) = \max_{\subseteq}\{R_1 \cap R_2 \mid < R_1, R_2 > \in \mathcal{R}^{\vee}_{\subseteq}(\{\psi_1, .., \psi_k\}, K) \times \mathcal{R}_{\subseteq}(\psi_{k+1}, K)\}$, for each $0 \leq k < l$. At each step, from the definition of $\mathcal{R}_{\subseteq}$, both $|\mathcal{R}^{\vee}_{\subseteq}(\{\psi_1, .., \psi_k\}, K)|$ and $|\mathcal{R}_{\subseteq}(\psi_{k+1}, K)|$ are bounded by $q(|K|)$, so the number of candidate intersections $R_1 \cap R_2$ to compute is bounded by $q(|K|)^2$, and the number of pairwise comparisons wrt set inclusion (for the $\max_{\subseteq}$ operation) is bounded by $q(|K|)^2 \cdot |\mathcal{R}^{\vee}_{\subseteq}(\{\psi_1, .., \psi_{k+1}\}, K)| < q(|K|)^3$. In addition, there are $l - 1 = |\Psi^i_K - 1| < |\Psi^i_K|$ iterative steps.

So if $f_2(n)$ expresses as a function of $|K|$ the cost of computing all $\mathcal{R}^{\vee}_{\subseteq}(\Psi^i_K, K)$ for all $1 \leq i \leq m$, then $f_2(n) = O(\sum_{i=1}^{m}(|\Psi^i_K| \cdot g(n) + |\Psi^i_K| \cdot (q(n)^2 n \log n + q(n)^3 n \log n))$. And because $\sum_{i=1}^{m} |\Psi^i_K| = |\Psi_K|$, this may be factorized into $f_2(n) = O(|\Psi_K| \cdot (g(n) + q(n)^2 n \log n + q(n)^3 n \log n)) = O(|\Psi_K| \cdot (g(n) + q(n)^3))$.

Finally, let us consider the computation of all $\mathcal{R}^{\vee}_{\subseteq}(\Psi^{>i}_K, K)$ for $1 \leq i < m$ (which was also temporarily assumed to be known). If $\mathcal{R}^{\vee}_{\subseteq}(\Psi^i_K, K)$ is known for each $1 \leq i \leq m$, for instance by applying the above procedure (i.e. the procedure whose cost is f_2), then each $\mathcal{R}^{\vee}_{\subseteq}(\Psi^{>i}_K, K)$ can be computed by induction on i, starting with

$i = m-1$. For the base case, we have $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{>m-1}, K) = \mathcal{R}^{\vee}_{\subseteq}(\Psi_K^m, K)$. Then for each $0 \leq i < m-2$, $\Psi_K^{>i} = \Psi_K^{>i+1} \cup \Psi_K^{i+1}$, such that from lemma 6.5.2.2.1, $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{>i}, K)$ can be computed as $\max_{\subseteq}\{R_1 \cap R_2 \mid < R_1, R_2 > \in \mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{>i+1}, K) \times \mathcal{R}_{\subseteq}(\Psi_K^{i+1}, K)\}$. The number of candidate intersections at each iterative step is once again bounded by $q(|K|)^2$, the number of pairwise comparisons wrt set-inclusion by $q(|K|)^3$, and there are m inductive steps. Let $f_3(n)$ express the cost as a function of $|K|$ of computing all $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{>i}, K)$ for all $0 \leq i < m$, provided $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{i}, K)$ has already been computed for all $1 \leq i \leq m$. Then $f_3(n) = O(m(q(n)^2 n \log n + q(n)^3 n \log n)) = O(mq(n)^3)$.

Therefore if $f_4(n)$ expresses the cost the execution of algorithm 2 as a function of $|K|$, then $f_4(n) = f_1(n) + f_2(n) + f_3(n) = O(mq(n)^3) + O(|\Psi_K| \cdot (g(n) + q(n)^3)) + O(mq(n)^3)$. And because $m \leq |\Psi_K|$, this can be reduced to $f_4(n) = O(|\Psi_K| \cdot q(n)^3) + O(|\Psi_K| \cdot (g(n) + q(n)^3)) + O(|\Psi_K| \cdot q(n)^3) = O(|\Psi_K| \cdot (3q(n)^3 + g(n)) = O(|\Psi_K| \cdot (q(n)^3 + g(n))$.

For phase 2 of the procedure, i.e. for algorithm 3, the explanation is almost identical, and therefore it will not be presented in full details here. The function Refine in particular is very similar to algorithm 2. The families $\mathcal{Q}$, $\mathcal{Q}_2$, $\mathcal{O}_2$, $\mathcal{H}_{def}$, and $\mathcal{H}'_{def}$ also have the property that their elements are maximal wrt set inclusion, therefore their cardinality is also bounded by $q(n)$.

So line 11 of function Refine, the number of generated intersections is bounded by $q(n)^2$, and the number of pairwise comparisons lines 12 and 13 by $q(n)^3$. For the verification line 18 that $\Psi_{Q_2}^{>i+1} \neq \emptyset$, if $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{>i+1}, K)$ is known (and the cost of its computaion has already been taken into account in $f_4(n)$), then the number of pairwise comparisons is bounded by $q(n)^2$. The cost of computing $\mathcal{V}$ has already been taken into account in $f_4(n)$ as well, so if $f_5(n)$ expresses as a function of $|K|$ the cost of the execution of one iteration of the main loop of function Refine, assuming that $\mathcal{V}$ and $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{>i+1}, K)$ are known, then $f_5(n) = O(q(n)^2 n \log n + q(n)^3 n \log n = O(q(n)^3)$.

The cardinality of $\mathcal{V}$ is not bounded by $q(n)$ though. Intuitively, $\mathcal{V}$ contains all base remainders sets for all $\psi \in \Psi_K^{i+1}$. For instance, it may be the case that $\mathcal{V} = \{< R_2, \psi_1 >, < R_2, \psi_2 >\}$. Therefore $|\mathcal{V}|$ is bounded by $|\Psi_K^{i+1}| \cdot q(n)$. Note that this does not affect the cardinality of $\mathcal{H}_{def}$. Line 7 of function UPDATE, it may be the case that a same intersection R_3 is generated multiple times during the execution. But because $\mathcal{H}_{def}$ is a set, R_3 can appear only once in it, and line 10, a single value $\mathcal{H}'(R_1) \cup \{\psi\}$ for $\mathcal{H}(R_3)$ is computed. So in a sense, the choice of this value is non-deterministic: if $\mathcal{V} = \{< R_2, \psi_1 >, < R_2, \psi_2 >\}$ for instance, then $\mathcal{H}(R_3)$ may take either $\mathcal{H}'(R_1) \cup \{\psi_1\}$ or $\mathcal{H}'(R_1) \cup \{\psi_2\}$. But as shown in section 6.5.1, and in particular by lemma 6.5.1.9, this does not affect the correctness of the procedure. So lines 4 and 5 of function UPDATE, $|\mathcal{H}'_{def}(R_1) \times \mathcal{V}| < q(n) \cdot (|\Psi_K^{i+1}| \cdot q(n)) = |\Psi_K^{i+1}| \cdot q(n)^2$. Line 6, because $\mathcal{H}'(R_1) \subseteq \Psi_K^{i+1}$ from lemma 6.5.1.4, $|\mathcal{H}'(R_1)| \leq |\Psi_K^{i+1}|$ must hold, such that the cost of verifying whether $\psi \in \mathcal{H}'(R_1)$ is in $O(|\Psi_K^{i+1}|)$. Line 7, the number of generated intersections is bounded by $|\mathcal{H}'_{def}(R_1) \times \mathcal{V}| < |\Psi_K^{i+1}| \cdot q(n)^2$. And line 8, $|\mathcal{H}'_{def}| < q(n)$. So the number of pairwise comparisons is bounded by $|\Psi_K^{i+1}| q(n)^2 \cdot q(n) = |\Psi_K^{i+1}| \cdot q(n)^3$. So if $f_6(n)$ expresses as a function of $|K|$ the cost of one execution of function UPDATE, then $f_6(n) = O(|\Psi_K^{i+1}| \cdot q(n)^2 n + |\Psi_K^{i+1}| \cdot q(n)^2 n \log n + q(n)^3 n \log n)) = O(|\Psi_K^{i+1}| \cdot q(n)^3)$.

Finally, let $f_7(n)$ express as a function of $|K|$ the cost of the execution of algorithm 3, provided $\mathcal{R}_{\subseteq}(\psi, K)$ is known for all $\psi \in \Psi_K$, and $\mathcal{R}^{\vee}_{\subseteq}(\Psi_K^{\geq i}, K)$ for all $0 \leq i < m$ (the cost of the computation of these is already taken into account in $f_4(n)$). Then $f_7(n) = O(\sum_{i=1}^{m}(f_5(n) + f_6(n))) = O(\sum_{i=1}^{m}(O(q(n)^3) + O(|\Psi_K^i| \cdot q(n)^3)))$, and because $\sum_{i=1}^{m} |\Psi_K^i| = |\Psi_K|$, we have $f_7(n) = O(O(q(n)^3) + O(|\Psi_K| \cdot q(n)^3)) = O(|\Psi_K| \cdot q(n)^3)$.

So $f(n) = f_4(n) + f_7(n) = O(|\Psi_K| \cdot (q(n)^3 + g(n)) + O(|\Psi_K| \cdot q(n)^3) = O(|\Psi_K| \cdot (2q(n)^3 + g(n)) = O(|\Psi_K| \cdot (q(n)^3 + g(n))$.

Then because $g(n) = \Omega(q(n))$, $q(n) = O(g(n))$ must hold. So $q(n)^3 = O(g(n)^3)$, such that $f(n) = O(|\Psi_K| \cdot (q(n)^3 + g(n)) = O(|\Psi_K| \cdot (g(n)^3 + g(n)) = O(|\Psi_K| \cdot g(n)^3)$.

Now if $c > 1$, $(c^n)^3 = c^{3n} = O(c^n)$ holds. So for any function h such that $h(n) = \Omega(c^n)$, $(h(n))^3 = O(h(n))$ must hold as well. In particular, from proposition 6.3.2.13, there is a $c > 1$ such that $g(n) = \Omega(c^n)$, so $(g(n))^3 = O(g(n))$ must hold, and because $f(n) = O(|\Psi_K| \cdot (g(n)^3)$, we have $f(n) = O(|\Psi_K| \cdot g(n))$.

Chapter 7

Introducing explicit ontological distinctions

After the detection of nonsense on a linguistic basis in a consistent/coherent KB K, addressed in Chapters 4 and 6, this chapter investigates the detection of nonsense in K on a formal basis, extending K with explicit ontological distinctions, in order to yield an inconsistent/incoherent KB. After this extension step, more traditional (and more or less automated) KB debugging methods may be applied to solve the inconsistency/incoherence, for instance computing justifications (see Chapter 3 Section 3.5), or performing syntax-based revision like described in Chapter 8.

A first requirement on this extension phase is that it should be meaningful, i.e. the inconsistency/incoherence of the extended KB should be caused by violations of common sense of the type described in Chapter 1, which are the primary target of debugging strategies developed throughout this thesis. To this end, the strategy proposed in this chapter relies on formal ontology (as a discipline), and more specifically on a foundational ontology (as an artifact), both introduced in Section 7.1. This is

an arguably unconventional use of foundational ontologies, which are traditionally used to structure a new KB rather than correcting it. Instead, the most influential debugging framework based on formal ontology is probably the OntoClean methodology [GW00], presented in Section 7.2. But applying OntoClean to a whole KB has been empirically shown to be complex and time-consuming, as explained in Section 7.2.3.

Section 7.3 proposes a shallow and partial, but robust and cost-efficient alternative. It is robust in that any (fragment of a) foundational ontology expressed in OWL may be used for that purpose. And it is cost-efficient because it relies on the manual attachment[1] to the foundational ontology of a very limited number of elements of $\text{sig}(K)$, based on the simple intuition that an element of $\text{sig}(K)$ is more likely to be used with incompatible meanings within K if it appears in a higher number of axioms.[2] The objective is to maximize the number of common sense violations potentially spotted while limiting the amount of manual work.

Finally, Section 7.4 proposes an evaluation of this extension strategy based on the foundational ontology TMEO [JVZ+14], which can be viewed as a simplified and slightly extended version of the taxonomy of DOLCE [MBG+03]. If K is the input KB, Θ the foundational ontology, and Δ the manually crafted attachment axioms, then this extension strategy is evaluated based on its ability to identify erroneous axioms of K as involved in the inconsistency/incoherence of $K \cup \Theta \cup \Delta$. An axiom $\phi \in K$ is involved in the inconsistency/incoherence of $K \cup \Theta \cup \Delta$ iff it appears in some justification (see Chapter 3 Section 3.5) for the inconsistency/incoherence. These first empirical results seem to indicate that a shallow and fast ontological analysis

[1] The meaning of *attachment* will be made more precise in Section 7.3.1.

[2] This formulation is a slight simplification for the sake of readability. In practice, not only is the number of axioms of K in which an element of $\text{sig}(K)$ appears taken into account, but also to a certain extend the diversity of these axioms, as explained in Section 7.3.2.

may be sufficient to identify a reasonable number of actually erroneous axioms as involved in the inconsistency/incoherence.

As an alternative to TMEO, one of the datasets used in Chapter 8 relies on the attachments of DBpedia concepts to the foundational ontology Proton [TKM05]. These attachments to Proton are not evaluated in Section 7.4, because they were performed by [DKSP10] independently from the work presented in this thesis.[3]

7.1 Formal ontology and foundational ontologies

Formal ontology, as a discipline, provides philosophically grounded axiomatizations of domain independent notions, for instance theories of parthood, of time and space, of identity, of qualities, etc., as well as high-level categorizations of "what there is".

From a more applied perspective, foundational ontologies, also called *top-level ontologies*, like BFO or DOLCE (both described in [MBG+03]), have been successfully used as the backbone of multiple KBs (e.g. CIDOC CRM [Doe03] for DOLCE), or even families of KBs (e.g. Bioportal[4] for BFO). They are prototypically based on a small set of primitive and relatively abstract unary predicates/DL atomic concepts such as `Physical Entities`, `Events`, `Time intervals`, etc....

Each foundational ontology reflects a specific conceptualization, and these conceptualizations tend to be mutually exclusive, corresponding to alternative modeling choices. As an illustration, the choice is made in DOLCE to reify qualities, inspired by trope theory [Cam81]. The example used by [MBG+03] to illustrate this choice is the color of a given flower, which, according to DOLCE, is considered as an individual, distinct from the flower itself, and from the value of this color. This individual

[3] Proton was nonetheless manually extended with a small number of disjointness axioms for these experiments, i.e. axioms of the form $A \sqsubseteq \neg B$, as explained in Chapter 8 Section 8.6.1.

[4] `http://bioportal.bioontology.org/`

can be predicated over (as it can be predicated over in natural language), and its value (hue $\times$ saturation $\times$ luminosity) may change with time. No assumption is made though as to whether such a conceptualization more accurately describes the nature of things (whatever this may mean) than another one which does not reify qualities. Therefore a given KB K may or may not adopt this view, depending on applicative requirements. The important point is that if K does (or does not) follow this view, it should do (or not do) it *systematically*.

Another interesting illustration was given by example 1.1.1, in Chapter 1. Some of these axioms are reproduced here, together with a few other ones, once again from DBpedia:

Ex 7.1.1.

$K = \{$

(1) $\texttt{occupation}(\textit{Peter Munk}, \textit{CEO})$

(2) $\top \sqsubseteq \forall \texttt{occupation}.\texttt{PersonFunction}$

(3) $\exists \texttt{hasPersonName}.\top \sqsubseteq \texttt{PersonFunction}$

(4) $\texttt{occupation}(\textit{Peter Munk}, \textit{Peter Munk 1})$

(5) $\texttt{occupation}(\textit{Ernest Noel}, \textit{Ernest Noel 1})$

(6) $\texttt{occupation}(\textit{Ernest Noel}, \textit{Ernest Noel 2})\}$

The intended meaning of *Peter Munk 1*, *Ernest Noel 1* and *Ernest Noel 2* in axioms 4 to 6 may not be obvious at first sight. It becomes clear though from the following DBpedia datatypeProperty assertions:[5]

(a) $\texttt{title}(\textit{Peter Munk 1}, \text{“CEO of Barrick Gold”})$

[5]The second argument of these assertions is a string, not a DL individual. In particular, it cannot be predicated over. For the treatment of datatypeProperty assertions adopted in this thesis, see Chapter 2 Section 2.3.1.

(b) `title`(*Ernest Noel 1*, "Member of Parliament")

(c) `title`(*Ernest Noel 2*, "Businessman")

Ernest Noel 1 stands for "Ernest Noel as a member of Parliament", and *Ernest Noel 2* for "Ernest Noel as a businessman", such that `occupation` in axioms 4 to 6 is understood as ranging over individual roles (similar to the qua-individuals of [MGV+05]) which are specific to a given human being. This meaning of `occupation` is reinforced by axiom 3, which suggests a mapping from such roles to person names. Representing individual roles in a given KB may or may not be relevant, depending on applicative requirements. But once again, if the choice is made to adhere to such a conceptualization, it should probably be done in a systematic fashion. In particular here, if the instances of `PersonFunction` are meant to be individual roles, and/or if `occupation` is understood as ranging over individual roles,[6] then the first statement is clearly an outlier.

Now a foundational ontology which commits to the representation of individual roles, will also commit to the fact that nothing can be both an individual role and a profession, title or activity like *CEO* (i.e., in formal ontology terms, a *role* [MVB+04]).[7] For instance, "Peter Munk as CEO of Barrick Gold", i.e. *Peter Munk 1* above, is certainly not the same individual as *CEO* understood as the professional activity of *Peter Munk*. In particular, other human beings could have *CEO* as an activity, whereas *Peter Munk 1* can be the occupation of *Peter Munk* only (in ontological terms, it is specifically dependent on *Peter Munk 1*).

So if Θ is a (fragment of a) foundational ontology where individual roles and ac-

[6] The disproportion in example 7.1.1 between the two meanings of `occupation` (ranging over either individual roles or activities) may not (and is not meant to) reflect its usage in DBpedia as a whole. In particular there are other occurrences of `occupation` similar to axiom (1).

[7] In order to avoid possible confusions, this meaning of the word *role* for formal ontology has nothing to do with its meaning in *DL role*.

tivities are modeled, and if `IndividualRole` and `Activity` are the two elements of $\text{sig}(\Theta)$ respectively representing these notions, then $\Theta \vdash \texttt{IndividualRole} \sqsubseteq \neg\texttt{Activity}$ will hold. Let us also assume that none of the predicates or constants of Θ appears in K, i.e. $\text{sig}(K) \cap \text{sig}(\Theta) = \emptyset$, which is generally the case. *CEO* in K apparently designates an activity, which can be expressed by $\texttt{Activity}(\mathit{CEO})$, whereas according to its dominant use within K, `occupation` ranges over individuals roles, which may be expressed by $\top \sqsubseteq \forall\texttt{occupation}.\texttt{IndividualRole}$. Provided a good understanding of Θ, and assuming the focus is already put on *CEO* and `occupation`, this type of manual analysis can be performed rapidly, by reviewing the respective occurrences of *CEO* and `occupation` in K. And extending $K \cup \Theta$ with $\Delta = \{\texttt{Activity}(\mathit{CEO}), \top \sqsubseteq \forall\texttt{occupation}.\texttt{IndividualRole}\}$ yields an inconsistent $K \cup \Theta \cup \Delta$.

In practice, a manual ontological analysis of the input KB with the help of a core ontological knowledge Θ may be more or less difficult to perform, depending on Θ. For instance, the OntoClean methodology, presented in the next section, relies on some arguably complex or subtle ontological distinctions. The notions of quality and individual role described above may also (to a lesser extent) be perceived as complex, or not commonly used outside of the formal ontology community (and DBpedia for the latter). But in practice, an efficient analysis can also be performed with more standard ontological distinctions. In particular, the formal ontology TMEO used in Section 7.4 (as well as Proton) relies on widely accepted categories in knowledge engineering, such as physical entities, events, time intervals, etc.

7.2 OntoClean

The OntoClean methodology [GW00] is probably the best-known KB debugging approach based on formal ontology, and differs by several aspects from the shallow ontological analysis just illustrated. Section 7.2.1 is an introduction to OntoClean as a theory, whereas Section 7.2.2 reviews its different OWL implementations. Finally, Section 7.2.3 explains the (practical) motivations behind the use of a shallow analysis with a top-level ontology instead of OntoClean for the experiments described in Section 7.4.

7.2.1 Overview

OntoClean provides a set of second-order constraints which should be satisfied by a taxonomy (i.e. by the set of consequences of a KB of the form $A \sqsubseteq B$, with A and B DL atomic concepts).

The approach is initially based on four metaproperties, namely *rigidity, identity, unity* and *dependence*, which may or may not be verified by each atomic concept of a KB.

The example of rigidity will be briefly developed in order to give a better understanding of the approach. An atomic concept A is said to designate a *rigid property* (in the OntoClean terminology) iff all instances of A are necessarily instances of A in some alethic modal sense. In other words, if e is an instance of a rigid property A in some admissible world W (actual, alternative, past, future, ...), then in any other admissible world W', either e is an instance of A in W', or e does not exist in W'. The converse holds as well: if for any instance e of A in some admissible world, $A(e)$ must hold in all other admissible worlds where e exists, then A is rigid.

For instance, according to most conceptualizations, a person cannot cease to be a person, neither can a wedding cease to be a wedding, such that the atomic concepts `Person` and `Wedding` in a given KB may be considered as rigid. A *non-rigid* property is a property which is not rigid, i.e. a property such that some of its instances can cease to instantiate it in some admissible world, and still exist. A specialization of non-rigidity is *anti-rigidity*: a property A is anti-rigid iff all its instances can cease to be instances of A in some admissible world and still exist. For instance, being a student or being unemployed are generally viewed as anti-rigid properties. For *non-rigid* but not anti-rigid properties, [GW02] provides the example of being hard, which is necessary for some instances (e.g. a hammer), but not for other ones (e.g. a sponge).

The three other meta properties are *identity, unity and dependence*, and will only be briefly introduced (most examples are taken from [GW02] or [GW09]).

For *identity*, a distinction is made between concepts which supply an identity criterion, carry it, or do not supply or carry any. For instance, one may consider that `Human being` supplies an identity criterion, because two human beings can be distinguished based on their fingerprints, and because having fingerprints is necessary for a human being, whereas `Red` does not carry or supply an identity criterion. Note that these identity criteria do not need to be expressed in the KB.

Unity characterizes concepts whose instances must be wholes, and such that all its instances are wholes in virtue of the same relation (not necessarily expressed in K) which unifies its parts. As an illustration, all instances of `Ocean` may be considered as wholes in virtue of a so-called *topological unifying relation*, and all instances of `Hammer` may be considered as wholes in virtue of a so-called *functional unifying relation*. On the other hand, `Legal Agent`, if it encompasses human beings and organizations, carries *no unity*, because its instances, even if they are wholes,

may have different unifying relations. Finally, `Amount of water` carries *anti-unity* because none of its instances must be a whole.

Dependence is the last of the four metaproperties in the original formulation of OntoClean. A concept A carries dependence if each instance of A implies the existence of another individual.

Together with these metaproperties, OntoClean provides a set of second-order axioms which constrain subsumption between atomic concepts. For instance, an anti-rigid atomic concept should only (transitively) subsume anti-rigid atomic concepts. Another example is the fact that an atomic concept with anti-unity should only (transitively) subsume atomic concepts with anti-unity.

Applying the OntoClean methodology consists in manually assigning metaproperties to the atomic concepts of an input KB K, and then check whether the taxonomy of K verifies the OntoClean constraints. For instance, let us assume that $\{\texttt{Person}, \texttt{Student}\} \subseteq N_{Con}(K)$, and that `Person` was assigned the metaproperty "rigid", and `Student` the metaproperty "anti-rigid". If $K \vdash \texttt{Person} \sqsubseteq \texttt{Student}$, the first constraint above is violated, which is likely to indicate a modeling error.

7.2.2 Implementations

At least three implementations of OntoClean have been developed for OWL. One of them is a plug-in for the Protege ontology editor,[8] referenced by [Wel06], but not documented online anymore. The two others are [Wel06] and [GRV10]. Both propose a form of meta-modeling, which, from a relatively abstract point of view, amounts to maintaining a theory $M = \mathrm{Cn}(M)$, where atomic concepts of K are reified, i.e. represented as individuals.

[8] http://protege.stanford.edu/

M is such that $\text{sig}(M) \cap \text{sig}(K) = \emptyset$, with a bijective mapping reif() from $N_{Con}(K)$ to $N_{Ind}(M)$. For instance, $\texttt{Person} \in N_{Con}(K)$ iff $\text{reif}(\texttt{Person}) \in N_{Ind}(M)$. A DL atomic role $\texttt{subClass} \in N_{Con}(M)$ represents the subsumption relation over the elements of N_{Con} according to K, i.e. $\texttt{subClass}(\text{reif}(A_1), \text{reif}(A_2)) \in M$ iff $K \vdash A_1 \sqsubseteq A_2$. $\text{sig}(M)$ also contains atomic concepts for OntoClean metaproperties, for instance $\texttt{Rigid} \in N_{Con}(M)$, and the fact that `Person` was assigned the metaproperty "rigid" is expressed with $\texttt{Rigid}(\text{reif}(\mathit{Person})) \in M$. Finally, OntoClean second order axioms are represented as first-order formulas ranging over reified concepts, for instance $\texttt{NonRigid} \equiv \neg\texttt{Rigid} \in M$, $\texttt{AntiRigid} \sqsubseteq \texttt{NonRigid} \in M$, and $\texttt{AntiRigid} \sqsubseteq \forall\texttt{subsumes}.\texttt{AntiRigid} \in M$.

The implementation of [GRV10] is an improvement over the one of [Wel06], in that K and M can be integrated as a unique KB K', such that $K \subseteq K'$ and $M \subseteq \text{Cn}(K')$, without compromising decidability, relying on some advanced features of OWL 2 /$\mathcal{SROIQ}$. In particular, even after a modification of the axioms of K within K', $K' \vdash A_1 \sqsubseteq A_2$ iff $K' \vdash \texttt{subClassOf}(\text{reif}(A_1), \text{reif}(A_2))$ still holds.

[GRV10] also implemented a largely ad hoc (for OntoClean) but interesting tracing mechanism, in order to identify the source of the violations of an OntoClean constraint, as an alternative to the potentially costly computation of all justifications (see Chapter 3 Section 3.5) for the inconsistency of K'.

It should be noted that these implementations do not express the whole OntoClean axiomatics. For instance, OntoClean requires that if $K \vdash A_1 \sqsubseteq A_2$ and A_2 carries an identity criterion, then A_1 must carry the same identity criterion. But identity criteria may not be represented in K, and the ontological analysis in these cases for [Wel06] and [GRV10] only consists in determining whether a given atomic concept supplies/carries an identity criterion, without the need to specify it.

7.2.3 Limitations

The most common criticisms of the OntoClean methodology are the cost and difficulty of performing a manual tagging of the atomic concepts of the input KB. In particular, as mentioned by [GW02], the identity and unity metaproperties are subtle notions, and for that reason may be difficult to understand or apply. Both are defined wrt to an additional condition (respectively an *identity criterion* and a *unifying relation*) which is in not expressed in general by K. For instance, what could be identity criterion of `PersonFunction` in example 7.1.1 is hard to pinpoint, but it is equally hard to decide that there is one. Therefore, as experienced by [VVSH08], inter-annotator agreement tends to be relatively low for real input datasets.

In addition, whether a given atomic concept in $N_{Con}\text{sig}(K)$ verifies a given metaproperty is often not obvious from K, even for the arguably simpler case of rigidity. For instance, in the illustration of the OntoClean methodology given by [GW09], the choice is made to assign anti-rigidity to the atomic concept `Food`, considering that nothing is essentially food, because it may never be eaten. But this is presented as a *choice* by [GW09], and arguably, other characterizations of `Food` may have been adopted. In the same vein, whether `Tool` should be considered as rigid, non-rigid or anti-rigid may not be explicit from a given K. Actually, one of the merits of formal ontology is to constrain a knowledge engineer to make such decisions, and formalize them explicitly. But in a debugging scenario, if the person who performs the ontological analysis is not the author of the KB, then assigning a metaproperty to a concept may be an overinterpretation.

Finally, the OntoClean method is limited to the identification of incorrect subsumptions between atomic concepts, i.e consequences of K of the form $A_1 \sqsubseteq A_2$, with A_1 and A_2 atomic concepts. This does not reduce debugging to the removal

or update of subsumption *axioms* between atomic concepts though. For instance, if $K \vdash A_1 \sqsubseteq A_2$, $\Gamma = \{A_1 \sqsubseteq \exists R.\top, \exists S.\top \sqsubseteq A_2, R \sqsubseteq S\}$ and $\Gamma \subseteq K$, then Γ is a possible explanation for $A_1 \sqsubseteq A_2$. But in practice, incorrect *ABox axioms* (see Chapter 2 Section 2.3.7) tend to go unnoticed when the OntoClean methodology is applied, because cases where an ABox axiom is involved in the derivation of a formula of the form $A \sqsubseteq B$ are infrequent, even though still possible for the more expressive DLs.

7.3 Shallow ontological analysis with a top-level

As explained in Chapter 1 Section 1.1.2, OWL expressible datasets published on the LOD cloud may contain sets of statements which are intuitively absurd, although logically consistent/coherent. In this case, a manual ontological analysis may be performed in order to yield an inconsistency/incoherence, and identify violations of common sense in such datasets on a logical basis.

The OntoClean methodology, described in the previous section, is an illustration of this strategy, but was shown to be costly and error-prone. This section describes an arguably basic but less costly alternative, which consists in attaching a selection of elements of the signature of the input KB K to a (DL expressible subset of a) foundational ontology Θ, which provides the core ontological knowledge required for the analysis, in the form of an axiomatization of domain independent first-order categories like `Event`, `PhysicalEntity`, etc.

The notion of attachment is made explicit in Section 7.3.1, whereas Section 7.3.2 proposes a way to select a subset of $\text{sig}(K)$. Section 7.3.3 discusses the advantages and drawbacks of such a fast and shallow analysis, and Section 7.3.4 lists some possible variations of this strategy, which are not evaluated in this thesis.

7.3.1 Attachments

If K is the consistent/coherent input KB and Θ a (consistent/coherent) additional theory, then $\text{sig}(K) \cap \text{sig}(\Theta) = \emptyset$ generally holds, such that for an inconsistency/ incoherence to be derived, some additional set Δ of "anchoring" formulas is needed, which relates $\text{sig}(K)$ and $\text{sig}(\Theta)$. For instance, when applying the OntoClean methodology as described above in Section 7.2.2, if Θ is the OntoClean axiomatization, Δ is composed of formulas of the form $B(\text{reif}(A))$, where B is a metaproperty like `AntiRigid`, and $\text{reif}(A)$ the reification of a predicate $A \in N_{Ind}(K)$.

If Θ is a first-order theory instead, Δ may be composed of formulas which will be called *attachments*, and simply defined as follows. If $e \in N_{Ind}(K)$, then an attachment for e is a formula δ of the form $B(e)$, with $B \in N_{Con}(\Theta)$. If $A \in N_{Con}(K)$, then an attachment for A is a formula δ of the form $A \sqsubseteq B$, with $B \in N_{Con}(\Theta)$. If $R \in N_{Role}(K)$, then an attachment for R is a formula δ of the form $\top \sqsubseteq \forall R.B$, $\exists R.\top \sqsubseteq B$ or $R \sqsubseteq S$, with $B \in N_{Con}(\Theta)$ or $S \in N_{Role}(\Theta)$. In addition, in all cases, B (resp. S) is the most or one of the most specific concepts (resp. roles) in $N_{Con}(\Theta)$ (resp. $N_{Role}(\Theta)$) which intuitively verifies δ. As an illustration, in example 7.1.1 above, `Activity`(CEO) and $\top \sqsubseteq \forall$`occupation.IndividualRole` are possible attachments for CEO and `occupation` respectively.

For a given $q \in \text{sig}(K)$, if no obvious candidate B (or S) can be found in $\text{sig}(\Theta)$, then no attachment is performed for q. Another particular case is the one where several candidates can be found in $\text{sig}(\Theta)$, but are incompatible according to Θ. For instance, in example 7.1.1, according to axiom (1), $\top \sqsubseteq \forall$`occupation.Activity` may be an alternative candidate attachment for `occupation`. In such cases, the prevailing meaning of q in K is chosen, or if no meaning obviously prevails, none is selected, i.e. none of the two incompatible attachments is made.

7.3.2 Selective attachments

In order to limit the amount of manual work, a selection of elements of $\text{sig}(K)$ may be attached only. A simple heuristic consists in prioritizing the attachment of elements of $\text{sig}(K)$ with a larger number of syntactic occurrences within K, based on the intuition that an individual or predicate is more likely to be used with incompatible meanings within K if it appears in a larger number of axioms. A selection $s(\text{sig}(K))$ of elements of $\text{sig}(K)$ to be attached can thus be performed on a simple syntactic basis.

Then for each $q \in s(\text{sig}(K))$, a brief manual review of the axioms in which q appears (with the possible help of annotations about q in K) is generally sufficient to identify the prevailing meaning of q within K, if there is one. For instance, in example 7.1.1, the prevailing meaning of `occupation` is apparently that of a relation holding between human beings and individual roles.

This heuristic may be refined though, in order to take into account the variety of the axioms in which each element r of $\text{sig}(K)$ appears. For instance, the individual *Boston* appears 539 times in DBpedia in an axiom of the form $\texttt{location}(e, \mathit{Boston})$, with e a named individual, but (very probably) every time with the same meaning. In order to take this (frequent) pattern into account, for the experiments described in Section 7.4, an additional heuristic is applied to compute $s(\text{sig}(K))$. For each $e_1 \in N_{Ind}(K)$, before computing the frequency of e_1 in K, K is quotiented by the equivalence relation $\sim$ defined by $\phi_1 \sim \phi_2$ iff $((\phi_1 = R(e_1, e_2)$ and $\phi_2 = R(e_1, e_3))$ or $((\phi_1 = R(e_2, e_1)$ and $\phi_2 = R(e_3, e_1)))$, such that for instance the axioms $\texttt{location}(\mathit{Wang\ Theater,\ Boston})$ and $\texttt{location}(\mathit{Ether\ Dome,\ Boston})$ count as one occurrence only for the individual *Boston*.

Then a maximum threshold of n elements to attach in each of $N_{Ind}(K)$, $N_{Con}(K)$

and $N_{Role}(K)$ can be applied, as an attempt to reduce the amount of manual work while maximizing the number of common sense violations to be spotted.

7.3.3 Advantages and drawbacks

It should first be noted that this shallow ontological analysis does not aim at reproducing or replacing an ontological analysis with OntoClean. In particular, the axioms of the input K involved in the inconsistency/incoherence may be different in both cases. They may also differ depending on the foundational ontology Θ being used. More generally, identifying all common sense violations within K with a single ontological analysis is very unlikely, and this should instead be viewed as an attempt to make some of them explicit.

A first practical advantage of a first-order foundational ontology, when compared to a second-order axiomatic like OntoClean, is that Δ does not need to predicate over reified DL concepts or DL roles, but instead can be constituted of simple DL formulas predicating over the domain modeled by K, for instance $\texttt{Boat} \sqsubseteq \texttt{PhysicalEntity}$, $\texttt{Event}(\textit{2016 US Presidential Elections})$, $\top \sqsubseteq \forall \texttt{causes}.\texttt{Event}$ or $\texttt{causes}.\top \sqsubseteq \texttt{Event}$. Predicating (and quantifying) over the individuals of the modeled domain is arguably easier than assigning second order properties to concepts (or relations). For instance, deciding whether a given boat is a physical entity, or whether instances of the concept `Boat` as it is understood in K are physical entities, is probably less controversial than deciding whether the concept `Boat` carries or supplies an identity criterion, or is a rigid concept.

Another advantage of choosing such a Δ is that ABox axioms within K are more likely to be involved in the inconsistency/incoherence of $K \cup \Theta \cup \Delta$, which in turn has two interesting consequences: ABox axioms can be used to identify errors within

the TBox/RBox, and erroneous ABox axioms may be spotted as well.

A potential risk of the approach though, which is also a risk of the OntoClean methodology (see section 7.2.3), is overinterpreting the input KB K. In particular, some distinctions may not be made in K, resulting in predicates or individuals in $\text{sig}(K)$ which are ambiguous wrt Θ. For instance, a city, region, state, country, etc. in DBpedia may generally designate either an administrative entity or a physical location, whereas the foundational ontology TMEO for instance, used in the following experiments, considers that nothing can be both. The difficulty in such cases is to determine for some $q \in s(\text{sig}(K))$ whether the prevailing meaning of q in K corresponds to one of two incompatible meanings according to Θ (and which one), or whether q is used ambiguously for both in K. The notion of prevailing meaning in K of some $q \in s(\text{sig}(K))$, although intuitive, is arguably difficult to formalize, especially when the size of K is limited, and even a shallow ontological analysis like this one may be performed differently by two annotators. For instance, in example 7.1.1, one may choose to consider that the size of K is not sufficient to decide that one of the two meanings of `occupation` (ranging over either individual roles or activities) is significantly predominant, in which case `occupation` may be viewed as inherently ambiguous wrt Θ.

As another illustration, the dataset $K_{1.1}^{DBP}$, introduced in Chapter 5 and used in Section 7.4, is a small subset of DBpedia, where the atomic concept `Place` is used 12 times, with the meaning of "physical place" in at least 10 of them. For this reason, the choice was made to attach it to the DL atomic concept `Location` of TMEO,[9] even though this may not be the prevailing meaning of `Place` within DBpedia.

Because overinterpretations, misinterpretations, or simply subjective interpretations remain a possibility, if some additional procedure is applied to solve the in-

[9] The attachments of elements of $\text{sig}(K_{1.1}^{DBP})$ to TMEO are reproduced in Appendix B.

consistency/incoherence resulting from the ontological analysis, like the procedure that will be presented in Chapter 8, a simple precaution consists in considering as potentially erroneous not only the axioms of K, but also the axioms of Δ.

Another practical limitation is obviously the size of the KB. Determining the prevailing meaning of each $q \in s(\text{sig}(K))$ (or determining that q it is ambiguous wrt Θ) requires reviewing all axioms in which q appears in K, which may still be costly for large KBs.

7.3.4 Alternatives

Different variations of this shallow ontological analysis may be performed as alternatives, and no claim is made here as to which strategy is the most efficient. In particular, if $q \in s(\text{sig}(K))$ is ambiguous in K wrt Θ, then attachment axioms may include disjunctions. For instance, in example 7.1.1, a possible attachment for the range of `occupation` would be $\top \sqsubseteq \forall \texttt{occupation}.(\texttt{IndividualRole} \sqcup \texttt{Activity})$. The probability of raising an inconsistency/incoherence by applying this strategy is lower, but the risk of overinterpreting is limited.

Alternatively, on may choose to make both attachments, regardless of the prevailing meaning or q. For instance, in example 7.1.1, both $\top \sqsubseteq \forall \texttt{occupation}.\texttt{Activity}$ and $\top \sqsubseteq \forall \texttt{occupation}.\texttt{IndividualRole}$ may be added to Δ, in which case one of the two will probably need to be discarded when solving the resulting inconsistency/incoherence.[10]

[10] From a logical point of view, a third solution consists in weakening K such that $K \cup \Delta \cup \Theta \vdash \top \sqsubseteq \forall \texttt{occupation}.\bot$), but this is clearly not satisfying.

7.4 Evaluation

This section presents an evaluation of the strategy just introduced. It illustrates the fact that a shallow and fast ontological analysis based on a (fragment of a) foundational ontology may nonetheless be efficiently used to identify violations of common sense within an input KB K.

7.4.1 Objective of the evaluation

The evaluation estimates to what extent actually erroneous axioms within K can be identified as involved in the inconsistency/incoherence of $K \cup \Theta \cup \Delta$, after an ontological analysis, with Θ the core ontological knowledge, and Δ the attachments (see Section 7.3.1 above) of elements of $\text{sig}(K)$ to Θ.

To this end, the family $\mathcal{J}$ of all justifications (see Chapter 3 Section 3.5) for the inconsistency/incoherence of $K \cup \Theta \cup \Delta$ is computed, i.e. $J \in \mathcal{J}$ iff it is a minimal inconsistent or incoherent subset of $K \cup \Theta \cup \Delta$ wrt $\subseteq$. Intuitively, an axiom ϕ of K is involved in the inconsistency/incoherence iff $\phi \in \bigcup \mathcal{J}$. In other words, if $E \subseteq K$ is the set of actually erroneous axioms within K, this evaluation is based on the proportion of elements of E which are present in $\bigcup \mathcal{J}$, i.e. on $\frac{|E \cap \bigcup \mathcal{J}|}{|E|}$.

It is important to understand that such an evaluation can be based on recall only. Indeed, if $J \in \mathcal{J}$, then $\frac{|E \cap J|}{|J|}$ does not provide meaningful information about the efficiency of the approach, because the number of axioms of $(K \setminus E) \cup \Theta \cup \Delta$ in J may vary independently. So precision defined as $\frac{|E \cap J|}{|\bigcup \mathcal{J}|}$ would not provide any useful indication. For instance, if $|E| = 10$, $|E \cap \bigcup \mathcal{J}| = 8$ and $|\bigcup \mathcal{J}| = 2000$, this is still a better result than if $|E| = 10$, $|E \cap \bigcup \mathcal{J}| = 7$ and $|\bigcup \mathcal{J}| = 10$, independently from $|\bigcup \mathcal{J}|$.

7.4.2 Datasets

Because the evaluation is based on recall, i.e. on $\frac{|E \cap \bigcup \mathcal{J}|}{|E|}$, $|E|$ must be known, and therefore the set of all actually erroneous axioms within K must have been previously identified. This is why two small subsets of DBpedia were used for this evaluation, which could be manually reviewed. These are $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$, both described in Chapter 5 Section 5.1, of 225 and 159 axioms respectively.

They were both independently reviewed by a formal ontology expert, who identified 11 and 7 erroneous axioms within $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$ respectively. This includes both ABox and TBox axioms, precisely 9 ABox an 2 TBox axioms for $K_{1.1}^{DBP}$, and 6 ABox and 1 TBox axiom for $K_{1.2}^{DBP}$.

A third and larger dataset is also used, namely K_{1}^{DBP} (8329 axioms), but only in order to evaluate how many attachment axioms are needed to obtain an inconsistency/incoherence.

7.4.3 Foundational ontology

The core ontological knowledge Θ for this evaluation is the ontology backing the categorization tool TMEO used in the Senso Comune project, in order to develop a lexical-ontological resource [JVZ+14]. It is a simplified and slightly extended variant of the top-level DOLCE [MBG+03], formalized as DL expressible first-order constraints.

TMEO can be viewed as a minimal foundational ontology for this task. It is composed of 69 axioms only, all of which are of the form $A_1 \sqsubseteq A_2$ (40 axioms) or $A_1 \sqsubseteq \neg A_2$ (29 axioms), with A_1 and A_2 atomic concepts. It signature is composed of 38 atomic concepts, which represent relatively abstract domain-independent categories, such as `Event`, `TangibleEntity`, `MentalEntity`, `Artifact`, etc. An OWL version

(with comments) is available online.[11] The taxonomy of TMEO is also reproduced in Appendix B Section B.1 for completeness.

Discussing the structure of TMEO is out the scope of this work though, and other foundational ontologies may have been used instead. In particular, some experiments run in Chapter 8 rely on a subset of DBpedia extended with the foundational ontology Proton, which is of a larger size, and also more expressive than TMEO. The attachments to Proton were performed independently from this work by [DKSP10], and not specifically in a debugging perspective, therefore they will not be evaluated in the current section. For more information about this fragment of Proton, the reader is referred to Chapter 8 Section 8.6.

7.4.4 Attachments

The set Δ of attachments of elements of $\text{sig}(K)$ to Θ for this evaluation is composed of manually crafted statements of the form described in Section 7.3.1 above, with the exception of $R \sqsubseteq S$ (with R and and S DL roles), because $N_{Role}(\text{TMEO}) = \emptyset$.

In order to select the set $s(\text{sig}(K))$ of candidate elements of $\text{sig}(K)$ for attachment, each of $N_{Ind}(K), N_{Con}(K)$ and $N_{Role}(K)$ was ordered by decreasing number of syntactic occurrences within K. This heuristic was refined by quotienting K before computing these occurrences, as explained in Section 7.3.2. An integer parameter n was used to test different sizes for Δ, such that, roughly speaking, if $\mathcal{M} = \{N_{Ind}(K), N_{Con}(K), N_{Role}(K)\}$,[12] then the n most frequent elements in K of each $M \in \mathcal{M}$ (modulo the above quotienting of K) were selected as candidate for attachment. This description of $s()$ is an approximation, and is sufficient for a global

[11] http://juliencorman.github.io

[12] For this specific experiment dataypeProperties were counted a DL roles, and not a DL atomic concepts, such that N_{Role} here actually designates the objectProperties and datatypeProperties appearing in K.

understanding of the evaluation protocol. The exact selection procedure is slightly more complex, in order to account for the fact that several elements of M may have identical frequencies. It is reproduced here for the sake of completeness only. Let $f(m)$ denote the frequency of m, let $\preceq_M$ be the total preorder over each $M \in \mathcal{M}$ defined by $m_1 \preceq_M m_2$ iff $f(m_1) \leq f(m_2)$, let $\sim_M$ be the equivalence relation defined over M by $\preceq_M$, and let $M^{\preceq_M} = (M_1^{\preceq_M}, .., M_{|M/\sim_M|}^{\preceq_M})$. The set of possible values for n is N, defined by $n \in N$ iff there is a $M \in \mathcal{M}$ and a $1 \leq i \leq |M/\sim_M|$ such that $|M_1^{\preceq_M} \cup .. \cup M_i^{\preceq_M}| = n$. Now within each $M \in \mathcal{M}$, let j be the largest integer such that $|M_1^{\preceq_M} \cup .. \cup M_j^{\preceq_M}| \leq n$. Then $M \cap s(\text{sig}(K)) = M_1^{\preceq_M} \cup .. \cup M_j^{\preceq_M}$.

For each element q of $s(\text{sig}(K))$ (individual, DL concept or DL role), attachments of the form described in Section 7.3.1 were added to Δ only if some obvious most specific concept(s) for q to be attached to could be found within N_{Con}(TMEO).

7.4.5 Results

For K_1^{DBP}, setting n to 2, with only 5 attachment axioms in Δ, was sufficient to obtain an inconsistent KB. This shows that with a very limited amount of work, it is possible to identify a probable violation of common sense within K, by focusing on the elements of sig(K) which are the most frequent in K.[13]

The results and statistics for $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$ are given in Table 7.1. Column "$|K|$" gives the number of axioms in the input KB, and column "$|E|$" the number of these axioms which were manually identified as erroneous. Column "$|\text{sig}(K)|$" gives the size of the signature of K, and column "$|s(\text{sig}(K))|$" the number of elements of sig(K) which were candidate for an attachment to Θ, based on the parameter n described above. Column "Att." gives the number of elements of $s(\text{sig}(K))$ which

[13] modulo the above heuristic for ABox axioms of the form $R(e_1, e_2)$

		$\|K\|$	$\|E\|$	$\|\text{sig}(K)\|$	$\|s(\text{sig}(K))\|$	Att.	$\|\Delta\|$	$\|E \cap \bigcup \mathcal{J}\|$	Rec.
K^{DBP}	$n = 6$	225	11	111	12	11	12	7	0.64
K^{DBP}	$n = 6$	159	7	73	15	11	15	2	0.29
	$n = 11$				30	23	29	7	1

Table 7.1: Shallow ontological analysis: results

were actually attached to Θ, i.e. for which a concept could be found in Θ to perform an attachment, and column "$|\Delta|$" gives the number of attachment axioms. Note that multiple attachments are possible for a given $q \in s(\text{sig}(K))$, in particular for DL roles (domain and range), which explains that the values in columns "Att." and "$|\Delta|$" may differ. Finally, column "$|E \cap \bigcup \mathcal{J}|$" gives the number of erroneous axioms in $\bigcup \mathcal{J}$, i.e. the number of erroneous axioms involved in the inconsistency/incoherence, and column "Rec." is the recall, i.e the value of $\frac{|E \cap \bigcup \mathcal{J}|}{|E|}$.

For $K^{DBP}_{1.1}$, setting $n = 6$ yielded 12 candidates for attachment, 11 of which could be attached to Θ, for a total of $|\Delta| = 12$ attachment axioms. This was sufficient to obtain $|E \cap \bigcup \mathcal{J}| = 7$, i.e. to identify 7 of the 11 erroneous axioms as potentially responsible for the inconsistency/incoherence. Increasing n did not provide better results though, i.e $|E \cap \bigcup \mathcal{J}| = 7$ still holds for $n > 6$, which is why no other value for n is reported in table 7.1.

For the second KB $K^{DBP}_{1.2}$, setting $n = 6$ was disappointing, with 15 candidates for attachment, 11 being attached and 15 attachment axioms, but only 2 erroneous axioms in $\bigcup \mathcal{J}$, over 7 $K^{DBP}_{1.2}$. But increasing n to 11 produced more satisfying results. The amount of manual work is slightly superior, with 30 elements of $K^{DBP}_{1.2}$ candidate for attachment, and 23 attached, for a total of $|\Delta| = 29$ attachment axioms. But this was sufficient to to cover all 7 erroneous axioms.[14]

[14] Only $n = 6$ and $n = 11$ were actually tested for $K^{DBP}_{1.2}$, and there were possible intermediate values for n, so 11 is an upper bound, i.e. there may be an n such that $6 < n < 11$ and n is sufficient to cover all erroneous axioms.

For each of K_1^{DBP}, $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$, the manual constitution of Δ with the above values for n took less than an hour. This seems to indicate that a relatively efficient shallow ontological analysis may indeed be performed with a very limited amount of manual work, by focusing on most frequent elements of the signature of K. All datasets and attachments are available online at `http://juliencorman.github.io`, as well as the manually identified erroneous axioms for $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$.

7.5 Conclusion

This chapter focused on the detection of nonsense in a consistent/coherent input KB on a logical basis, as an alternative to the linguistic input used in the preceding chapters. The solution under investigation consists in using already formalized explicit ontological distinctions, provided by formal ontology (as a research field). In other words, it amounts to committing temporarily to some external ontological view, and checking the compliance of an input KB K with this view.

If Θ is the external ontological knowledge, such an approach requires an additional set Δ of manually crafted axioms, which relate elements of the signature of K to elements of the signature of Θ, such that, intuitively, a violation in K of the constraints expressed by Θ should result in the inconsistency (or incoherence) of $K \cup \Theta \cup \Delta$.

The most influential illustration of this approach is probably the OntoClean methodology, which relies on a set of second-order constraints, and requires a manual assignment of second-order properties to the DL atomic concepts of K. OntoClean has been empirically shown to be costly and error-prone though, with a potentially low inter-annotator agreement, and is probably limited in practice to the identification of erroneous TBox axioms. OntoClean was introduced in Section 7.2, as well as

its OWL implementations, and their limitations.

As an alternative, a shallow and fast ontological analysis has been proposed in Section 7.3, which relies on an external (first-order) foundational ontology, with a focus on the elements of the signature of K with higher syntactic frequency. This proposal is primarily guided by practical feasibility, and arguably basic. But Section 7.4 empirically showed that this may actually be sufficient to identify absurd (TBox or ABox) axioms of K as involved in the inconsistency or incoherence of $K \cup \Theta \cup \Delta$, with a very limited amount of manual work.

Chapter 8

Syntax-based contraction/revision

This chapter focuses on the automatic detection of (a) set(s) of axioms to be preferably discarded or amended within and inconsistent or incoherent input KB. The initial motivation was the incorporation of linguistic evidence to the process, but most proposals made in this chapter go beyond this specific case, and can be viewed as more general contributions to the field of syntax-based contraction/revision, as introduced in Chapter 3 Section 3.6.2, which is itself closely related to diagnosis.

Section 8.1 characterizes the problem, and is partly a reminder of notions introduced in Chapter 3. It also identifies two of the main issues faced by syntax-based contraction/revision. The first one is the number of candidate output bases (or equivalently the number of diagnoses), and the second issue is computational cost.

Section 8.2 reviews solutions proposed in the literature to address these two issues, showing their respective limitations. The approach investigated here in order to deal with the first issue is a specific form of prioritized base debugging, defined in section 8.2. Intuitively, provided a preference relation (total preorder) $\preceq_a$ over the axioms of the input KB, it consists in prioritizing the removal of least preferred axioms wrt $\preceq_a$

until consistency (or coherence) is reached, together with discarding minimal sets of axioms wrt to set inclusion.

Section 8.3 discusses algorithmic aspects of syntax-based contraction/revision in more details, and in particular it clarifies the relation between the computation of minimal conflicts for an input KB on the one hand, and all base remainders/diagnoses on the other hand, based on the empirical observation that efforts to compute the former in the DL literature seem to have been more succesfull this far than efforts to compute the latter.

For black-box techniques, it is shown that both problems are strongly dependent, and more exactly that it cannot be guaranteed that all remainders/diagnoses have been computed with Reiter's algorithm if all minimal conflicts have not been computed as well during the execution, and that the converse holds, i.e. it cannot be guaranteed that all minimal conflicts have been computed if all remainders/diagnoses have not been computed as well.

For glass-box algorithms, although both problems are as hard computationally in the worst case, Section 8.3.2 investigates the conditions in which computing one of them may be more costly in practice than computing the other for the same input KB, which may be a reason why efforts to compute all justifications seem to have been more successful thus far than efforts to compute all base remainders/diagnoses.

Section 8.4 proposes an algorithm which performs prioritized base debugging as defined in section 8.2.2, provided the family of justifications for the inconsistency (or incoherence) is known, without the need to compute all base remainders.

Section 8.5 then addresses the obtention of the preference relation $\preceq_a$ over the axioms of the input KB, and more specifically over the axioms involved in the inconsistency. Assuming once again that the family $\mathcal{J}$ of justifications for the incon-

sistency/incoherence is known,[1] it defines a generic framework for evaluating the impact of discarding or retaining each potentially faulty axiom $\phi \in \bigcup \mathcal{J}$, in the form of two bases S_ϕ and $S_{\setminus\phi}$, which can be computed in time polynomial in $\sum_{J \in \mathcal{J}} |J|$,[2] and represent the part of the input KB which would necessarily be preserved if ϕ was respectively retained or discarded. These two bases, considered as theories, can then be evaluated in order to rank all potentially faulty axioms, i.e. in order to obtain $\preceq_a$. In particular, S_ϕ and $S_{\setminus\phi}$ may be evaluated based on their compliance to a linguistic corpus, as defined in Chapter 4 section 4.2.5. This possibility is evaluated in section 8.6.

Proofs are grouped in Section 8.8 for readability.

8.1 Characterization

This section is an introduction to the problems of syntax-based contraction and revision, also called belief base contraction and revision, or sometimes simply KB debugging. Most notions presented here have already been briefly introduced in Chapter 3 Section 3.6.2.

Section 8.1.1 characterizes the input and the task, and section 8.1.2 introduces a running example. Section 8.1.3 defines a series of central notions in order to understand the problem and its treatment here, most of which have already been introduced in Chapter 3. Finally, Section 8.1.4 briefly identifies two potential issues which are inherent to the task, namely the number of candidate outputs, and computational cost.

[1] or a reduction $\mathcal{V}$ of $\mathcal{J}$, introduced in Section 8.4, which verifies $|\mathcal{V}| \leq |\mathcal{J}|$, and for each $V \in \mathcal{V}$, there is a $J \in \mathcal{J}$ such that $V \subseteq J$.

[2] or $\sum_{V \in \mathcal{V}} |V| \leq \sum_{J \in \mathcal{J}} |J|$

8.1.1 Input and task

The input KB is assumed to be a (finite) set of DL formulas, and part of this KB may need to be preserved. The (possibly empty) part of the input KB to be preserved will be designated with Θ, and the part of the input KB to be weakened will be designated with K, such that the input KB is $K \cup \Theta$, with $K \cap \Theta = \emptyset$. For instance, if the input is a manually extended KB as described in Chapter 7, then Θ would correspond to the foundational ontology, and K to the rest of the KB (i.e. $K \cup \Delta$, according to the notation used in Chapter 7).

It is assumed that $K \cup \Theta$ is inconsistent, incoherent, or more generally that a set of undesired consequences of $K \cup \Theta$ has been previously identified. The problem consists in discarding axioms of K in order to restore the consistency/coherence of $K \cup \Theta$, or to get rid of the undesired consequences. For the sake of readability though, an important reduction of scope is made throughout this chapter: the focus will be put by default on the case of inconsistency, unless explicitly mentioned. Some authors [QLB06, QHH+08] actually distinguish revision wrt inconsistency from revision wrt to incoherence, proposing specific algorithms for each task. But all proposals made in this chapter can be adapted to the case of incoherence, or more generally undesired consequences of $K \cup \Theta$, provided minor modifications, so this distinction is not useful here.

The problem of discarding axioms from K in order to restore consistency[3] has been studied both in the fields of belief base revision/contraction and KB debugging (usually with $\Theta = \emptyset$ in the latter case). Again, in order to avoid possible confusions, belief base revision/contraction should be distinguished from belief set revision/contraction, whose most influential framework (the AGM framework [AGM85])

[3] or coherence, or more generally in order to get rid of undesired consequences

focuses on (deductively closed) theories, abstracting from syntax (see Chapter 3 Section 3.6 for a brief introduction to belief change). On the contrary, belief base revision/contraction requires the output to be a syntactic subset of $K \cup \Theta$, adapting several notions from the belief set revision/contraction literature, but with very different implications.

A comprehensive series of base revision and contraction operators applicable to DLs have been defined in [RW09], and part of the terminology used in this chapter is borrowed from that work. Precisely, the operations of interest here are named partial meet and kernel base revision without negation (with weak or full success) in [RW09]. A simplified notation will be used though, and most limit and/or trivial cases will not be explicitly addressed, in order to focus on the practical problem at hand. In particular, the set of statements Θ to be preserved is supposed to be consistent.

As explained in Chapter 3 Section 3.6.2.1, when the focus is on belief bases (i.e. sets of axioms) and not on belief sets (i.e. theories), the distinction between revision and contraction tends to blur: revising K by Θ is equivalent to contracting $K \cup \Theta$ by $\top \sqsubseteq \bot$, together with preserving Θ, which, as already discussed, does not hold for belief sets.

8.1.2 Example

The following running example will be used throughout this chapter.

Ex 8.1.1. $K = \{$

(1) `owningCompany`(*Smithsonian Networks*, *Smithsonian Institution*),

(2) `publisher`(*Birds of South Asia*, *Smithsonian Institution*),

(3) `occupation`(*Montgomery C. Meigs*, *Smithsonian Institution*),

(4) award(*James Dewar*, *Smithsonian Institution*),

(5) $\top \sqsubseteq \forall$owningCompany.Company,

(6) Company $\sqsubseteq$ Agent,

(7) $\top \sqsubseteq \forall$occupation.PersonFunction,

(8) $\top \sqsubseteq \forall$award.Award,

(9) Agent $\sqsubseteq$ AgentiveEntity,

(10) PersonFunction $\sqsubseteq$ NonAgentiveEntity,

(11) Award $\sqsubseteq$ NonAgentiveEntity }

$\Theta \vdash$ AgentiveEntity $\sqsubseteq \neg$ NonAgentiveEntity

Axioms 1 to 8 are DBpedia statements. Θ is a (fictional here) foundational ontology, and in this case the part of the input knowledge to be preserved. Axioms 9 to 11 are possible attachments of DBpedia classes to this top-level ontology. They correspond to the "anchoring" set Δ of axioms described in Chapter 7. So the DL atomic concepts AgentiveEntity and NonAgentiveEntity belong to sig(Θ), and all other individuals and predicates belong to the signature of DBpedia. The task here consists in discarding (a) subset(s) of axioms 1 to 11, in order to restore consistency. As opposed to the notation adopted in Chapter 7, K designates here all axioms which may be discarded, i.e. K in this chapter corresponds to $K \cup \Delta$ in Chapter 7. This is for more generality: the input of the debugging process presented in this chapter may be any inconsistent KB $K \cup \Theta$, whether it has been manually extended or not.[4] And again, it may also be the case that $\Theta = \emptyset$.

For instance, discarding axiom 1 is sufficient to get rid of the inconsistency, such that $(K \setminus (1)) \cup \Theta$ is a candidate output. Discarding axiom 9 is another possibility, as

[4]In order to be consistent with the notation adopted in Chapter 7, another option would be to systematically mention a possibly empty Δ throughout this chapter. But as will soon appear, this would seriously affected readability.

well as discarding axioms 3 and 8 for instance (but discarding 3 only or 8 only does not solve the inconsistency). On the other hand, discarding 1, 3, 8 and 9 altogether may be viewed as an unnecessary information loss.

8.1.3 Useful notions

This section is for a large part a reminder of some of the notions introduced in Chapter 3, presented here in more details and/or in a more formal way.

8.1.3.1 Hitting set

If $\mathcal{L}$ is the logic at hand, let $\text{hs} : 2^{2^{\mathcal{L}}} \mapsto 2^{2^{\mathcal{L}}}$ be the function which returns all *hitting sets* for an input family of sets of formulas, i.e.:

Definition 8.1.3.1. *hitting sets*
$\text{hs}(\mathcal{X}) = \{\Gamma \subseteq \bigcup \mathcal{X} \mid \forall X \in \mathcal{X} : \Gamma \cap X \neq \emptyset\}$

According to the notation introduced in Chapter 2 Section 2.2, if all elements of $\mathcal{X}$ are finite, then $\min_{\subseteq}(\text{hs}(\mathcal{X}))$ returns the elements of $\text{hs}(\mathcal{X})$ which are minimal wrt $\subseteq$, i.e. $\Gamma \in \min_{\subseteq}(\text{hs}(\mathcal{X}))$ iff $\Gamma \in \text{hs}(\mathcal{X})$ and for all $\Gamma' \in \text{hs}(\mathcal{X})$, $\Gamma' \not\subset \Gamma$.

This notion will be used to establish the correspondence between justifications and remainders, for the case where $\mathcal{X}$ is a finite family of finite subsets of $\mathcal{L}$.

In order to avoid possible confusions, it should be noted that computing $\min_{\subseteq}(\text{hs}(\mathcal{X}))$ is not identical to the canonical (intractable) *minimal hitting set* problem (which is itself equivalent to the covering set problem) studied in the optimization literature. Given a finite family of finite sets $\mathcal{X}$, the canonical minimal hitting set problem consists in computing *one* minimal hitting set *wrt to cardinality*, i.e. one element of $\{Y \mid |Y| \in \min_{Z \in \text{hs}(\mathcal{X})} |Z|)\}$, whereas $\min_{\subseteq}(\text{hs}(\mathcal{X}))$ is the family of *all* minimal hitting

sets *wrt set inclusion*. Computing $\min_{\subseteq}(\text{hs}(\mathcal{X}))$ is not tractable either though, which can simply be seen from the fact that the size of a solution to this problem may be exponential in $n = |\bigcup \mathcal{X}|$. For instance, if $\mathcal{X}$ is a partition of $\bigcup \mathcal{X}$ with $|X| = 2$ for all $X \in \mathcal{X}$, then $|\min_{\subseteq}(\text{hs}(\mathcal{X}))| = 2^{|\mathcal{X}|} = 2^{\frac{n}{2}}$.

8.1.3.2 Base remainders

$\mathcal{R}$ will designate the family of all subsets of K consistent with Θ, i.e.:

Definition 8.1.3.2. $\mathcal{R} = \{R \subseteq K \mid R \cup \Theta \nvdash \bot\}$

If Γ is a finite set of DL formulas and Ψ is a set of DL formulas, [RW09] define a *generalized base remainder* for Ψ (understood disjunctively) in Γ as a maximal subset R of Γ wrt $\subseteq$ such that $\text{Cn}(R) \cap \Psi = \emptyset$, and the *generalized base remainder set* as the family of all generalized base remainders. The generalized base remainder set corresponds to the construction designated with $\mathcal{R}^{\vee}_{\subseteq}(\Psi, \Gamma)$ in Chapter 4 Section 6.3.2.2.1, i.e. $R \in \mathcal{R}^{\vee}_{\subseteq}(\Psi, \Gamma)$ iff $\text{Cn}(R) \cap \Psi = \emptyset$ and for all $R \subset R' \subseteq \Gamma$, $\text{Cn}(R') \cap \Psi \neq \emptyset$. Intuitively, a base remainder is a candidate output base for a contraction of Γ by Ψ (understood disjunctively).

For the (base) revision of K by Θ in DLs, this notion needs to be slightly adapted: $\mathcal{R}^{\vee}_{\subseteq}(\{\top \sqsubseteq \bot\}, K \cup \Theta)$ is not adequate, or in other words Γ above cannot be replaced by $K \cup \Theta$, because this does not account for the fact that Θ should be preserved. If the negation $\overline{\Theta}$ of Θ is defined, then a solution consists in contracting K by $\overline{\Theta}$, and then extending it with Θ, or in other words, each candidate output is $R \cup \Theta$ such that $R \in \mathcal{R}^{\vee}_{\subseteq}(\overline{\Theta}, K)$. But most DLs do not allow for full negation (let alone full negation of a set of formulas). So in the rest of this chapter, the term *remainder set* will designate the family of maximal subsets of K consistent with Θ, and in order to simplify notation, without ambiguity here, it will be simply designated with $\mathcal{R}_{\subseteq}$:

Definition 8.1.3.3.

$\mathcal{R}_{\subseteq} = \{R \subseteq K \mid R \cup \Theta \nvdash \bot$ and for all $\phi \in (K \setminus R)$, $R \cup \{\phi\} \cup \Theta \vdash \bot\}$

Or equivalently:

Definition 8.1.3.4. $\mathcal{R}_{\subseteq} = \{R \in \mathcal{R} \mid$ for all $\phi \in (K \setminus R)$, $R \cup \{\phi\} \notin \mathcal{R}\}$

A strong intuition here is that if $R \in \mathcal{R}_{\subseteq}$, then $R \cup \Theta$ is a candidate output, and discarding any additional axiom can be viewed as an unnecessary information loss.

But there may be several elements in $\mathcal{R}_{\subseteq}$. For instance, in example 8.1.1, there are 12 elements in $\mathcal{R}_{\subseteq}$ (the exhaustive list of elements of $\mathcal{R}_{\subseteq}$ will be given in Section 8.1.3.4, as diagnoses, i.e. as their respective complements in K).

If this happens, in order to avoid non-determinism, a first solution consists in setting the output of the process to be the intersection $\bigcap \mathcal{R}_{\subseteq}$ of all remainders, then extended with Θ.[5] The resulting KB may still be disappointingly weak though. For instance, in example 8.1.1, $\bigcap \mathcal{R}_{\subseteq} \cup \Theta = \{2\} \cup \Theta$.

Another option was proposed in [MLB05], and consists in setting the output to be the disjunction of all remainders extended with Θ, designated here with $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$. This last construction is not natively representable in DLs, but can be simulated as a multibase, requiring that a formula be a consequence of $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$ iff it is a consequence of each remainder extended with Θ, i.e. $\mathrm{Cn}(\bigvee \mathcal{R}_{\subseteq} \cup \Theta) = \bigcap_{R \in \mathcal{R}_{\subseteq}} \mathrm{Cn}(R \cup \Theta)$.

The following observation is immediate (a proof is nonetheless provided in Section 8.8.1.1, for the sake of completeness):

Proposition 8.1.3.1. $\mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \bigcap_{R \in \mathcal{R}_{\subseteq}} \mathrm{Cn}(R \cup \Theta)$

[5] It is important to note that this construction is not equivalent to the so-called *full meet revision* defined for belief sets in the AGM framework (see Chapter 3 Section 3.6.1).

Therefore $\mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \mathrm{Cn}(\bigvee \mathcal{R}_{\subseteq} \cup \Theta)$. To see that the converse is not true, i.e. that $\mathrm{Cn}(\bigvee \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta)$ does not hold in general, consider the following counterexample, where $\mathcal{R}_{\subseteq} = \{R_1, R_2\}$ and $\Theta = \emptyset$, with $R_1 = \{A_1 \sqsubseteq A_2, A_3 \sqsubseteq A_4\}$ and $R_2 = \{A_1 \sqsubseteq A_2, A_3 \sqsubseteq A_5\}$. If $\psi = A_3 \sqsubseteq A_4 \sqcup A_5$, then $R_1 \vdash \psi$ and $R_2 \vdash \psi$, such that $\psi \in \bigcap_{R \in \mathcal{R}_{\subseteq}} \mathrm{Cn}(R \cup \Theta) = \mathrm{Cn}(\bigvee \mathcal{R}_{\subseteq} \cup \Theta)$. But $\bigcap \mathcal{R}_{\subseteq} \cup \Theta = \{A_1 \sqsubseteq A_2\} \nvdash \psi$, and so $\psi \notin \mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta)$.

Therefore the disjunction $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$ seems intuitively a better option than the intersection $\bigcap \mathcal{R}_{\subseteq} \cup \Theta$, in that it respects determinism as well, but with a smaller information loss. From a practical point of view though, depending on the engineering constraints, it may be quite unrealistic. There are at least two reasons for this:

- $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$ may not be natively representable as a single base in some common DLs (like $\mathcal{ALCH}$), and therefore it must be manipulated as a "family of bases whose corresponding theory is the intersection of the theories corresponding to each base", which is clearly not compatible with interoperability requirements for SW data, based on the publication and import of sets of axioms, not of family of sets of axioms.

- As an artifact, $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$ is stored and manipulated as the family $\mathcal{R}_{\subseteq}$, which requires the prior computation of $\mathcal{R}_{\subseteq}$. But as will be explained in section 8.3, in practice, computing $\mathcal{R}_{\subseteq}$ may be problematic.

It should finally be noted that $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$, even if stronger than $\bigcap \mathcal{R}_{\subseteq} \cup \Theta$, still tends to be disappointingly weak. For instance, in example 8.1.1, aside from axiom 2, none of the axioms of K is a consequence of $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$.

8.1.3.3 Selection function

In order to give up less information, it may be desirable to select some remainders, and yield as an output the intersection (or disjunction) of these selected remainders extended with Θ, or even submit them to the user if their number is small enough.

The notion of a *selection function* σ, once again adapted from the belief set revision literature, formalizes this idea. A selection function must select a nonempty subset of the remainder set. The requirement that σ selects strictly more than one remainder does not hold for belief bases, but only for belief sets, as explained in Chapter 3 Section 3.6.2.1. Therefore for the problem addressed here, σ could (and should ideally) select only one remainder, provided this choice is motivated.

Aside from their intuitive appeal, a good argument for the notions of (base) remainder and selection function is the representation theorem given in [RW09]. It states that for any selection $\sigma(\mathcal{R}_{\subseteq})$ of a nonempty subset of $\mathcal{R}_{\subseteq}$, the operators which take K and Θ and yields $\bigcap \sigma(\mathcal{R}_{\subseteq}) \cup \Theta$ satisfies a set of very intuitive rationality postulates for base revision (namely inclusion, weak consistency, strong success, pre-expansion and relevance), and conversely.

8.1.3.4 Diagnoses and incisions

A diagnosis is the complement in K of a base remainder. If $K \cup \Theta$ is the input KB, with Θ the subset to be preserved, in the general case where Ψ is a set of undesired consequences, a diagnosis D for K, Θ and Ψ (understood disjunctively) is a minimal set of axioms wrt $\subseteq$ such that $\mathrm{Cn}((K \setminus D) \cup \Theta) \cap \Psi = \emptyset$.

For the problem at hand here, K and Θ are unique, and $\Psi = \{\top \sqsubseteq \bot\}$, so for readability, the family of diagnoses for K, Θ and $\{\top \sqsubseteq \bot\}$ will simply be designated in this chapter by $\mathcal{D}$, defined as follows:

Definition 8.1.3.5. $D \in \mathcal{D}$ iff $D \subseteq K$, $(K \setminus D) \cup \Theta \nvdash \bot$, and for all $D \subset D' \subseteq K$, $(K \setminus D') \cup \Theta \vdash \bot$

Equivalently, $\mathcal{D}$ can be defined wrt $\mathcal{R}_{\subseteq}$ by:

Lemma 8.1.3.1. $\mathcal{D} = \{K \setminus R \mid R \in \mathcal{R}_{\subseteq}\}$

As an illustration, in example 8.1.1, there are 12 alternative diagnoses (and therefore also 12 base remainders), namely:
$\mathcal{D} = \{\{1\}, \{3,4\}, \{3,8\}, \{3,11\}, \{4,7\}, \{4,10\}, \{5\}, \{6\}, \{7,8\}, \{7,11\}, \{8,10\}, \{9\}\}$

The term *incision* will designate any superset of a diagnosis in K, i.e. an incision Γ is a subset of K such that $(K \setminus \Gamma) \cup \Theta \nvdash \bot$, but it may not be minimal wrt $\subseteq$. Equivalently, each incision is the complement in K of some $R \in \mathcal{R}$.

8.1.3.5 Justifications

If Γ is a finite set of DL formulas and Ψ a set of DL formulas, a justification J for Ψ (understood disjunctively) in Γ is a minimal subset of Γ wrt $\subseteq$ such that $\mathrm{Cn}(J) \cap \Psi \neq \emptyset$.

Again, in order to simplify notation, in the rest of this chapter, $\mathcal{J}$ will designate the family of all justifications for $\{\top \sqsubseteq \bot\}$ in $K \cup \Theta$, i.e.:

Definition 8.1.3.6. $J \in \mathcal{J}$ iff $J \subseteq K \cup \Theta$, $J \vdash \bot$, and for all $J' \subset J$, $J' \nvdash \bot$

In addition, the family $\mathcal{V}$ will be defined as follows:

Definition 8.1.3.7. $\mathcal{V} = \min_{\subseteq}\{J \cap K \mid J \in \mathcal{J}\}$

In other words, $\mathcal{V}$ is composed of all justifications for $\{\top \sqsubseteq \bot\}$ in $K \cup \Theta$, but reduced to their respective intersection with K, and minimal wrt $\subseteq$. Equivalently, $\mathcal{V}$ can be defined as the family of minimal subsets of K inconsistent with Θ, i.e.:

Definition 8.1.3.8. $V \in \mathcal{V}$ iff $V \subseteq K$, $V \cup \Theta \vdash \bot$, and for all $V' \subset V$, $V' \cup \Theta \nvdash \bot$

Note that $|\mathcal{V}| < |\mathcal{J}|$ may hold, because it may be the case that $J_1 \cap K \subseteq J_2 \cap K$ for $J_1, J_2 \in \mathcal{J}$ and $J_1 \neq J_2$. Note also that in the case where $\Theta = \emptyset$, $\mathcal{V} = \mathcal{J}$.

As an illustration, in example 8.1.1, there are 2 elements in $\mathcal{V}$, namely:
$\mathcal{V} = \{\{1, 3, 5, 6, 7, 9, 10\}, \{1, 4, 5, 6, 8, 9, 11\}\}$

Then the following theorem, adapted (among others) from [QHH+08], establishes the correspondence between $\mathcal{V}$ and $\mathcal{D}$:

Theorem 8.1.3.1. $\mathcal{D} = \min_{\subseteq}(\text{hs}(\mathcal{V}))$

Intuitively, in order to restore consistency in $K \cup \Theta$ while preserving Θ, it is sufficient (and necessary) to discard one axiom from each $V \in \mathcal{V}$, and discarding any additional axiom can be viewed as an unnecessary information loss.

An equivalent way of viewing this relation between $\mathcal{V}$ and $\mathcal{D}$ consists in representing $\mathcal{V}$ and $\mathcal{D}$ as boolean formulas. Let ρ be an injective mapping from $\bigcup \mathcal{V}$ to a set of propositional variables, let $\tau(\mathcal{V}) = \bigwedge_{V \in \mathcal{V}} (\bigvee_{\phi \in V} \rho(\phi))$, and let $\eta(\mathcal{D}) = \bigvee_{D \in \mathcal{D}} (\bigwedge_{\phi \in D} \rho(\phi))$. Then $\eta(\mathcal{D})$ is $\tau(\mathcal{V})$ in prime implicant form, and $\tau(\mathcal{V})$ is $\eta(\mathcal{D})$ in prime implicate form. So in theory, any of $|\mathcal{V}|$ and $|\mathcal{D}| = |\mathcal{R}_{\subseteq}|$ could be at most exponential in the size of each other.

In addition, because the elements of $\mathcal{V}$ are maximal wrt $\subseteq$, for all $V_1, V_2 \in \mathcal{V}$, if $V_1 \neq V_2$, then $V_1 \not\subseteq V_2$. So from the observation made in Chapter 6 Section 6.5.2.1, if $n = |\bigcup \mathcal{V}|$, then $|\mathcal{V}| \leq \binom{n}{\lceil \frac{n}{2} \rceil}$. Similarly, the elements of $\mathcal{D}$ are minimal wrt to set inclusion, such that for all $D_1, D_2 \in \mathcal{D}$, if $D_1 \neq D_2$, then $D_1 \not\subseteq D_2$ must hold. Therefore if $m = |\bigcup \mathcal{D}|$, then $|\mathcal{D}| \leq \binom{m}{\lceil \frac{m}{2} \rceil}$. And from theorem 8.1.3.1, $\bigcup \mathcal{D} \subseteq \bigcup \mathcal{V}$, so $m \leq n$, such that $|\mathcal{D}| \leq \binom{n}{\lceil \frac{n}{2} \rceil}$ also holds. Finally, from lemma 8.1.3.1, each $D \in \mathcal{D}$ is the complement in K of some $R \in \mathcal{R}_{\subseteq}$, so $|\mathcal{R}_{\subseteq}| = |\mathcal{D}|$, and $|\mathcal{R}_{\subseteq}| \leq \binom{n}{\lceil \frac{n}{2} \rceil}$ holds as well.

8.1.4 Practical limitations

8.1.4.1 Number of candidate output bases

A first practical limitation of KB debugging based on diagnosis is the number $|\mathcal{D}| = |\mathcal{R}_{\subseteq}|$ of alternative diagnoses/remainders, or equivalently the number of candidate output subbases (each candidate output being $R \cup \Theta$ for some $R \in \mathcal{R}_{\subseteq}$).

In the absence of a selection function (or equivalently if the selection function returns all remainders), the number of alternative solutions may be too high for them to be reviewed manually. And as illustrated by example 8.1.1 above, taking $\bigcap \mathcal{R}_{\subseteq} \cup \Theta$ as an output instead, or even $\bigvee \mathcal{R}_{\subseteq} \cup \Theta$, may yield a disappointingly weak KB (or a disappointingly weak disjunctive KB in the latter case).

As a further illustration, a modified version of Reiter's algorithm (described in Section 8.3.1) was implemented in order to compute a lower bound on the cardinality of $\mathcal{R}_{\subseteq}$, for the extended versions of datasets K_1^{DBP}, $K_{1.2}^{DBP}$ and $K_{1.3}^{DBP}$ presented in Chapter 7. This yielded $|\mathcal{R}_{\subseteq}| \geq 174$, $|\mathcal{R}_{\subseteq}| \geq 340$ and $|\mathcal{R}_{\subseteq}| \geq 270$ respectively, which is already too high for a manual review.

The solution to this problem adopted in this chapter is called *prioritized base revision*, and is formally characterized in section 8.2.2. Given a preference relation $\preceq_a$ over the axioms of K, a selection function σ is defined which intuitively prioritizes the removal of least preferred axioms, until consistency is reached.

8.1.4.2 Computational cost

Another important practical limitation of KB debugging based on diagnosis is its computational cost. Proposition 6.3.2.13 in Chapter 4 showed that in the worst case, computing $\mathcal{R}_{\subseteq}$ (or equivalently $\mathcal{D}$) has a cost at least exponential in $|K|$, regardless

of the underlying DL.

An interesting comparison can also be made between the respective costs of computing $\mathcal{J}$ and $\mathcal{R}_\subseteq$ (or equivalently $\mathcal{D}$). In the case where $\Theta = \emptyset$ (or more generally, when $\mathcal{J} = \mathcal{V}$), and if a glass-box algorithm is used, then as explained in Chapter 3 Section 3.5.1.2, computing $\mathcal{J} = \mathcal{V}$ or computing $\mathcal{R}_\subseteq$ are both equally hard wrt worst-case complexity. If α is the boolean pinpointing formula for the inconsistency of K, then computing $\mathcal{R}_\subseteq$ amounts to compute all prime implicates of α, and $\mathcal{J}$ its prime implicants. The relation between both tasks (computing $\mathcal{J}$ and computing $\mathcal{R}_\subseteq$) is even tighter for black-box algorithms. To our knowledge, black-box algorithms to compute the whole family $\mathcal{J} = \mathcal{V}$ of justifications [KPSH05, Hor11] rely on (a simplified version of) Reiter's algorithm, which, as shown in section 8.3.1, has the particularity that $\mathcal{D}$ can be obtained almost immediately during its execution as well. So if $\Theta = \emptyset$, computing $\mathcal{R}_\subseteq$ should not be harder (in the worst case for glass-box algorithms, in any case for black-box algorithms) than computing $\mathcal{J} = \mathcal{V}$.

If $\Theta \neq \emptyset$ instead, let $\mathcal{R}'_\subseteq$ be the family of base remainders for $\{\top \sqsubseteq \bot\}$ in $K \cup \Theta$, i.e. all maximal consistent subsets of $K \cup \Theta$ wrt $\subseteq$, or following the notation used in Chapter 6, $\mathcal{R}'_\subseteq = \mathcal{R}^\vee_\subseteq(\{\top \sqsubseteq \bot\}, K \cup \Theta)$. Then according to the above observation, computing $\mathcal{R}'_\subseteq$ should not be harder (in the worst case for glass-box algorithms, in any case for black-box algorithms) than computing $\mathcal{J}$. In addition, the following is immediate from the definitions of $\mathcal{R}_\subseteq$ and $\mathcal{R}'_\subseteq$ (a proof is nonetheless provided in section 8.8.1.2):

Proposition 8.1.4.1. $\mathcal{R}_\subseteq = \{R' \cap K \mid R' \in \mathcal{R}'_\subseteq \text{ and } \Theta \subseteq R'\}$

Checking whether $\Theta \subseteq R'$ holds and computing $R' \cap K$ are both in $O(|K \cup \Theta| \log |K \cup \Theta|)$, and as shown in Chapter 6 Section 6.5.2.1, $|\mathcal{R}'_\subseteq|$ is exponential in the worst case in $|K \cup \Theta|$, so computing $\mathcal{R}_\subseteq$ can be reduced to computing $\mathcal{R}'_\subseteq$ (as far

as worst-case complexity is concerned), and therefore it should not be harder than computing $\mathcal{J}$ either.

But in practice, attempts to compute remainders (or equivalently diagnoses) for DLs have been less successful than attempts to compute justifications. To our knowledge, most strategies proposed for DLs to compute diagnoses/remainders yield a subset only of $\mathcal{R}_{\subseteq}$ [Sch05, FS05, Kal06, QHH+08, CRW13], and/or rely on the computation of $\mathcal{J}$ beforehand [QHH+08, RW08, CRW13], with the exception of [MLBP06], but their experiments were limited to atomic concept unsatisfiability in TBoxes, and in relatively small KBs, or with low expressiveness.[6]

This observation may legitimately come as a surprise, given the above considerations. Computing $\mathcal{J}$ (or $\mathcal{V}$ in the case $\Theta \neq \emptyset$) beforehand in particular may at first sight seem inefficient. Because each $D \in \mathcal{D}$ is the complement in K of some $R \in \mathcal{R}_{\subseteq}$ and conversely, computing $\mathcal{R}_{\subseteq}$ and computing $\mathcal{D}$ are basically the same task. But from theorem 8.1.3.1, computing $\mathcal{D}$ out of $\mathcal{J}$ (resp. out of $\mathcal{V}$ in the case $\Theta \neq \emptyset$) amounts to computing all minimal hitting sets for $\mathcal{J}$ (resp. for $\mathcal{V}$), which is intractable from the observation made in section 8.1.3.1. And as just explained, computing $\mathcal{D}$ directly from $K \cup \Theta$ is not harder in theory than computing $\mathcal{J}$. So a case could be made for computing $\mathcal{D}$ (or equivalently $\mathcal{R}_{\subseteq}$) directly, without the need for $\mathcal{J}$. Another way to view this (in the case $\Theta = \emptyset$) is the following. If α is the pinpointing formula for $\{\top \sqsubseteq \bot\}$ in K, then computing $\mathcal{J}$ and then $\mathcal{D}$ out of $\mathcal{J}$ amounts to converting α into prime implicant form α', before converting α' into prime implicate form α'', instead of turning α directly into α''.

A (hypothetical) explanation may be suggested for this, which pertains to glass-box approaches only: from an empirical perspective, section 8.3.2 shows why computing the whole $\mathcal{J}$ with a glass-box approach may be more realistic in practice than

[6] in particular, ignoring disjointness axioms for larger input KBs

computing the whole remainder set $\mathcal{R}_\subseteq$ (or equivalently the whole family $\mathcal{D}$ of diagnoses), although worst-case complexity is identical for both tasks. Another possible reason is that most works which produce remainders (or equivalently diagnoses) out of justifications do not aim at computing the whole remainder set $\mathcal{R}_\subseteq$, but instead a selection $\sigma(\mathcal{R}_\subseteq)$ of remainders,[7] as will be illustrated in Section 8.2, and $\mathcal{J}$ (or $\mathcal{V}$) in these cases may be a relevant intermediate representation to compute $\sigma(\mathcal{R}_\subseteq)$. The algorithm described in section 8.4, which to our knowledge is a original proposal, actually falls into this category. It computes a selection $\sigma(\mathcal{R}_\subseteq)$ of remainders out of $\mathcal{J}$ (or out of $\mathcal{V}$ if $\Theta = \emptyset$), but without the need to compute the whole remainder set $\mathcal{R}_\subseteq$.

8.2 State of the art

This section reviews different solutions proposed in the literature in order to address the two issues which have just been identified. Most of the works reviewed in this section focused on the case where $\Theta = \emptyset$, which implies that $\mathcal{V} = \mathcal{J}$.

8.2.1 Structural heuristics

A first straightforward strategy to select a subset of $\mathcal{R}_\subseteq$ consists in selecting only the set $\mathcal{R}_\leq$ of elements of $\mathcal{R}$ with maximal cardinality. Obviously, $\mathcal{R}_\leq \subseteq \mathcal{R}_\subseteq$, so each $R \in \mathcal{R}_\leq$ is indeed a remainder. For DLs, the computation of (some elements of) $\mathcal{R}_\leq$ has been investigated by [FS05] in the case neither $\mathcal{V}$ nor $\mathcal{R}_\subseteq$ is known, using Reiter's algorithm (presented in section 8.3.1), and by [QHH+08] in the case $\mathcal{V}$ only is known.

[7] or sometimes a selection of elements of $\mathcal{R}$, some of which may not be remainders

But the rationale behind this selection function remains questionable, especially for relatively large datasets: for instance, if $|K| > 1000$, one may arguably wonder why discarding 12 axioms instead of 13 (or even 20) is necessarily preferable. This heuristic is actually used as a baseline in section 8.6, which shows that it may fail to discard the expected axioms. This is illustrated by example 8.1.1 above, where the best element of $\mathcal{R}_{\subseteq}$ is intuitively $K \setminus \{3, 4\}$, but it is not retained in $\mathcal{R}_{\leq}$, because there are 3 elements in $\mathcal{R}_{\subseteq}$ with higher cardinality. This heuristic also seems to be empirically biased towards the removal of TBox axioms rather than ABox axioms.

As an alternative, the notion of *core* was proposed in [SC03]. The authors define a core of arity n as a set of axioms which appear in n elements of $\mathcal{V}$, the intuition being that an axiom appearing in a core of large arity is more likely to be faulty. Both [QHH+08] and [CRW13] implemented a base contraction/revision strategy based on this notion, in the case where $\mathcal{V}$ is already known. It consists in iteratively discarding axioms which appear in the largest number of elements of $\mathcal{V}$ not hit thus far, until consistency is reached. For instance, if $\mathcal{V} = \{\{\phi_1, \phi_2\}, \{\phi_1, \phi_3\}, \{\phi_4, \phi_5\}\}$, then ϕ_1 is discarded first, because it appears in 2 elements of $\mathcal{V}$, against only 1 for each other axiom of $\bigcup \mathcal{V}$. Following this strategy, the family of computed incisions in this example is $\{\{\phi_1, \phi_4\}, \{\phi_1, \phi_5\}\}$. This strategy is actually the well-studied (optimal) approximation to the canonical minimal hitting set problem (wrt cardinality) mentioned in section 8.1.3.1. If $\mathcal{R}_{core}$ designates the complements of these incisions, this means that $\mathcal{R}_{core}$ and $\mathcal{R}_{\leq}$ are most of the time identical, yielding the same criticism as for $\mathcal{R}_{\leq}$. For instance, in example 8.1.1, $\mathcal{R}_{core} = \mathcal{R}_{\leq}$. But because it is only an approximation, an incision Γ obtained this way is not guaranteed to be minimal wrt cardinality among all incisions, i.e. $K \setminus \Gamma \notin \mathcal{R}_{\leq}$ may hold, or in other words, it is possible that $\mathcal{R}_{core} \not\subseteq \mathcal{R}_{\leq}$, as noted by [FS05]. A stronger observation can actually be made

here, namely that $\mathcal{R}_{core} \not\subseteq \mathcal{R}_{\subseteq}$ may hold as well.[8] To see this, consider the following example: $\mathcal{V} = \{\{\phi_1, \phi_2, \phi_3\}, \{\phi_1, \phi_4, \phi_5\}, \{\phi_1, \phi_6, \phi_7\}, \{\phi_2, \phi_4\}, \{\phi_3, \phi_6\}, \{\phi_5, \phi_7\}\}$. No element of $\mathcal{V}$ is a subset of another, so this is a possible configuration. As ϕ_1 appears in 3 elements of $\mathcal{V}$, its removal will be prioritized, and $\Gamma = \{\phi_1, \phi_3, \phi_4, \phi_7\}$ is one of the resulting incisions. But $\Gamma \setminus \{\phi_1\}$ is itself an incision, so Γ is not a minimal incision, or in other words $\Gamma \notin \min_{\subseteq}(\text{hs}(\mathcal{V}))$, and therefore $(K \setminus \Gamma) \notin \mathcal{R}_{\subseteq}$, such that $(K \setminus \Gamma) \cup \Theta$ could be extended with ϕ_1 without compromising consistency, i.e. ϕ_1 was unnecessarily discarded.

8.2.2 Prioritized base contraction/revision

Prioritized base contraction/revision formalizes the simple idea that all axioms within $K \cup \Theta$ are not equal, or in other words, that some preference relation $\preceq_a$ (total preorder, i.e. intuitively a ranking) over the axioms of $K \cup \Theta$ is available, such that if $\phi_1 \prec_a \phi_2$, the removal of ϕ_1 should be prioritized whenever possible when trying to restore the consistency of $K \cup \Theta$. The concrete way $\preceq_a$ may be obtained is discussed in section 8.5.

According to the notation adopted in Chapter 2 Section 2.2, for any $\Gamma \subseteq K \cup \Theta$, $(\Gamma_1^{\preceq_a}, .., \Gamma_n^{\preceq_a})$ will designate the ordered partition of Γ defined by $\preceq_a$.

[QHH+08] investigated the use of $\mathcal{V}$ in order to perform prioritized base revision, by computing all minimal incisions over the elements of $\mathcal{V}$ previously reduced to their lower-ranked axioms wrt $\preceq_a$. In other words, if $V_1^{\preceq_a}$ designates the equivalence class of least preferred axioms of $V \in \mathcal{V}$, and if $\mathcal{B} = \min_{\subseteq} \text{hs}(\{V_1^{\preceq_a} \mid V \in \mathcal{V}\})$, then the procedure selects $\mathcal{R}_m \subseteq \mathcal{R}$, defined by $\mathcal{R}_m = \{K \setminus B \mid B \in \mathcal{B}\}$. Each $B \in \mathcal{B}$ is such that $B \cap V \neq \emptyset$ for all $V \in \mathcal{V}$, therefore B is an incision, and so

[8] $\mathcal{R}_{core} \not\subseteq \mathcal{R}_{\subseteq}$ is stronger than $\mathcal{R}_{core} \not\subseteq \mathcal{R}_{\leq}$ because $\mathcal{R}_{\leq} \subseteq \mathcal{R}_{\subseteq}$, such that if $\Gamma \in \mathcal{R}_{core} \setminus \mathcal{R}_{\subseteq}$, then $\Gamma \in \mathcal{R}_{core} \setminus \mathcal{R}_{\leq}$.

each $R \in \mathcal{R}_m$ is consistent with Θ, i.e. $\mathcal{R}_m \subseteq \mathcal{R}$ holds. In addition, within each $V \in \mathcal{V}$, the removal of axioms with lowest preference is indeed prioritized. But a limit of this approach is (once again) that $\mathcal{R}_m \subseteq \mathcal{R}_\subseteq$ may not hold, as illustrated by the following example: $\mathcal{V} = \{V, V'\}$, with $V = \{\phi_1, \phi_2\}$, $V' = \{\phi_2, \phi_3\}$, and $\phi_1 \prec_a \phi_2 \prec_a \phi_3$. Then $V_1^{\preceq_a} = \{\phi_1\}$ and $V_1'^{\preceq_a} = \{\phi_2\}$, so the only possible incision is $B = \min_\subseteq \mathrm{hs}(\{V_1^{\preceq_a}, V_1'^{\preceq_a}\}) = \{\phi_1, \phi_2\}$, and therefore $\mathcal{R}_m = \{\{\phi_3\}\}$. But $\{\phi_1, \phi_3\} \cup \Theta \not\vdash \bot$, so ϕ_1 has been unnecessarily discarded, i.e. $\{\phi_3\} \notin \mathcal{R}_\subseteq$.

Another proposal is the so-called *lexicographic approach* [BCD+93], whose corresponding disjunctive KB can be computed (in propositional logic) with the *disjunctive maxi adjustment* procedure [BKLBW04]. This yields the family $\mathcal{R}_{\preceq_q} = \max_{\preceq_q} \mathcal{R}$,[9] with $\preceq_q$ a total preorder over $\mathcal{R}$ defined by:

Definition 8.2.2.1. $\preceq_q$

- $R \preceq_q R'$ iff $R' \not\prec_q R$
- $R \prec_q R'$ iff there is a $1 \leq i \leq n$ such that $|R_i^{\preceq_a}| < |R_i'^{\preceq_a}|$, and for all $1 \leq j < i$, $|R_j^{\preceq_a}| = |R_j'^{\preceq_a}|$

In this case too, the output is (the disjunction of) a selection of remainders, or in other words (a proof is provided in Section 8.8.2.2):

Proposition 8.2.2.1. $\mathcal{R}_{\preceq_q} \subseteq \mathcal{R}_\subseteq$

But this option is not completely satisfying either, because it is based on cardinality just like $\mathcal{R}_\leq$ above,[10] therefore yielding the same criticism.

[9] or a single base equivalent to the disjunction of these subbases in the case of *disjunctive maxi adjustment*.

[10] Actually, $\mathcal{R}_\leq$ is a specific case of $\mathcal{R}_{\preceq_q}$, where $\preceq_a$ defines only two equivalence classes, namely $\bigcup \mathcal{V}$ and $(K \cup \Theta) \setminus \bigcup \mathcal{V}$

Alternatively, assuming $\mathcal{V}$ is known, [Kal06] proposed to compute all minimal incisions which maximize the sum of the rankings of their respective axioms according to $\preceq_a$.[11] Because these incisions are minimal wrt $\subseteq$, the output is a set of remainders. But *ceteris paribus*, this heuristic again favors smaller sets of axioms wrt cardinality. For instance, let $\bigcup \mathcal{V} = \{\phi_1, \phi_2, \phi_3, \phi_4\}$, with $(\{\phi_1, \phi_2, \phi_3\}, \{\phi_4\})$ the ordered partition of $\bigcup \mathcal{V}$ defined by $\preceq_a$. If $r(\phi_i)$ is the ranking of axiom ϕ_i, with rank 1 for the preferred axioms wrt $\preceq_a$, then $r(\phi_4) = 1$, and $r(\phi_1) = r(\phi_2) = r(\phi_3) = 2$. Let $D_1 = \{\phi_1\}$ and $D_2 = \{\phi_2, \phi_3\}$ be two diagnoses. Then according to this heuristic, $K \setminus D_1$ will be preferred to $K \setminus D_2$ based on cardinality only, because $r(\phi_1) = 2 < r(\phi_2) + r(\phi_3) = 4$.

Finally, prioritized base revision has been characterized by [Neb92] with a (partial) order $\preceq_r$ over $\mathcal{R}$, defined by:

Definition 8.2.2.2. $\preceq_r$

- if $R =_r R'$, then $R = R'$,
- $R \prec_r R'$ iff there is $1 \leq i \leq n$ such that $R_i^{\preceq_a} \subset R_i'^{\preceq_a}$, and for all $1 \leq j < i$, $R_j^{\preceq_a} = R_j'^{\preceq_a}$

According to this view, the set of optimal subbases of K is the family $\mathcal{R}_{\preceq_r}$ defined by:

Definition 8.2.2.3. $\mathcal{R}_{\preceq_r} = \max_{\preceq_r} \mathcal{R}$

This solution is intuitively very similar to the solution $\mathcal{R}_{\preceq_q}$ presented above, but arguably more satisfying in that it is based on set inclusion instead of cardinality. Another practical difference is that $\preceq_q$ is a total order over $\mathcal{R}$, whereas $\preceq_r$ is a partial one.

The following observation is proven in section 8.8.2.1:

[11] or the sum of their scores if a $\preceq_a$ was defined by scores attributed to each axiom

Proposition 8.2.2.2. $\mathcal{R}_{\preceq_r} \subseteq \mathcal{R}_{\subseteq}$

In other words, each $R \in \mathcal{R}_{\preceq_r}$ is a base remainder, or equivalently, there is a selection function σ such that $\mathcal{R}_{\preceq_r} = \sigma(\mathcal{R}_{\subseteq})$.

This last solution is the one adopted in what follows. In particular, Section 8.4 provides an algorithm which computes $\mathcal{R}_{\preceq_r}$ out of $\mathcal{V}$, but without the need to compute the whole remainder set $\mathcal{R}_{\subseteq}$, which to our knowledge is an original proposal.

8.3 Minimal conflicts VS diagnoses: algorithms

This section compares the respective costs of computing $\mathcal{V}$ and computing $\mathcal{D}$ (or equivalently $\mathcal{R}_{\subseteq}$) with algorithms proposed in the literature for both tasks. Section 8.3.1 reproduces Reiter's algorithm as presented by [Was00], which to our knowledge is the most standard black-box technique used in the DL literature to compute either $\mathcal{V}$ or $\mathcal{D}$. It will also be shown that at each execution step of this algorithm, it cannot be guaranteed that $\mathcal{D}$ has actually been computed, unless $\mathcal{V}$ has been computed as well, and that the converse holds, i.e. it cannot be guaranteed that $\mathcal{V}$ has actually been computed, unless $\mathcal{D}$ has been computed as well.

Section 8.3.2 then focuses on so-called glass-box approaches, which are based on a saturated tableau for DLs, introduced in Chapter 3 Section 3.5.1.2. A detailed description of the algorithm is provided, together with an example of execution of a tableau for consistency in the DL $\mathcal{ALC}$. Although computing $\mathcal{V}$ or $\mathcal{D}$ with a saturated tableau are both as hard in the worst case, identifying the respective sources of a potential blowup for each task may suggest that the former (computing $\mathcal{V}$) could be a more realistic scenario in practice,

8.3.1 Black-box: Reiter's algorithm

Reiter's algorithm was originally proposed in [Rei87] (and corrected in [GSW89]) in order to identify faulty components within a system with abnormal behavior. The link between diagnoses and belief base contraction/revision was formally established in [Was00], and algorithm 6 is an almost literal transcription of the description of Reiter's algorithm given in [Was00].

In the rest of this section, any $\Phi \subseteq K$ such that $\Phi \cup \Theta \vdash \bot$ will be called a *conflict* for K and Θ, such that $\mathcal{V}$ is the family of all minimal conflicts for K and Θ.

The algorithm proceeds by expanding in a breadth-first fashion a directed acyclic graph (DAG) $\mathcal{G} = < \mathcal{N}, \mathcal{E}, \text{lab} >$, with $\mathcal{N}$ the set of nodes and $\mathcal{E} \subseteq \mathcal{N}^2$ the set of edges of $\mathcal{G}$, and $\text{lab} : \mathcal{N} \cup \mathcal{E} \mapsto 2^K$. $\mathcal{G}$ has a unique root node N_{root}. Each node N is labeled with a subset $\text{lab}(N)$ of K, such that $\text{lab}(N)$ is either a conflict, or $\emptyset$ by convention.[12] Each edge E is labeled with an axiom $\text{lab}(E)$ of K, and $h(N)$ will designate the set of axioms labeling the edges of a path from N_{root} to N. As shown in Section 8.8.3.1 (lemma 8.8.3.1), the procedure guarantees that all paths from N_{root} to N have the same set of labels, therefore $h(N)$ is determined for each N. At any step of the process, $\text{lab}(N) = \emptyset$ iff $h(N)$ is an incision for $\mathcal{V}$, i.e. iff $h(N)$ has a nonempty intersection with all elements of $\mathcal{V}$ (note that in this case, because $\mathcal{V}$ is the family of all minimal conflicts for K and Θ, $h(N)$ has also a nonempty intersection with all conflicts for K and Θ).

Algorithm 6 is a high-level description of the procedure.[13] F is a FIFO stack

[12] Alternative descriptions of the algorithm can be found in the literature, in particular expanding a tree, and not only a DAG. A conventional symbol is also often used instead of $\emptyset$ to indicate that a node is not labeled with a conflict.

[13] This is on purpose: a more detailed pseudo-algorithm would require records to represent nodes succession, and two hash tables for h and lab. But these are not needed for the points to be discussed in this section, and would simply hinder readability.

for the breadth-first expansion of the graph, initialized with $\{N_{root}\}$. The function ComputeConflict(X) returns a (not necessarily minimal) conflict $\Phi \subseteq X$ if there is one (i.e. if $X \cup \Theta \vdash \bot$), otherwise it returns $\emptyset$ by convention. The function CreateNode returns a new node. The procedure SetSuccessor (N_1, N_2, ϕ) creates an edge labeled with ϕ from node N_1 to node N_2, i.e. N_2 becomes the (unique) ϕ-successor of N_1, or more formally, $\mathcal{E}$ is replaced by $\mathcal{E} \cup \{(N_1, N_2)\}$, and $\text{lab}(N_1, N_2) = \{\phi\}$. The procedure DeleteEdge(N, ϕ) deletes from $\mathcal{E}$ the edge from N to its ϕ-successor. This last operation may trim the graph, i.e. some nodes (including N itself) may not be (transitively) successors of N_{root} after the application of this procedure, in which case they are discarded from the graph. More formally, discarding N from the graph consists in replacing $\mathcal{N}$ by $\mathcal{N} \setminus \{N\}$, and $\mathcal{E}$ by $\mathcal{E} \setminus \{(N_1, N_2) \mid N_1 = N \text{ or } N_2 = N\}$. The procedure DeleteOrphanNodes(F) updates the stack F accordingly, deleting from F any node discarded from the graph, i.e. any node which is not (transitively) a successor of N_{root} anymore. Finally, conditional statements like "If there is a N_2 s.t." or "If there is a N_3 s.t.", should be understood as "If there is a node N_i in the graph such that", and N_i is considered "in the graph" iff $N_i \in \mathcal{N}$ iff N_i is (transitively) a successor of N_{root}, i.e. if there is a path from N_{root} to N_i.

Let $\mathcal{G}^0, .., \mathcal{G}^n$ be the successive states of $\mathcal{G}$ after i executions of the main loop of Algorithm 6, such that $\mathcal{G}^0 =< \{N_{root}\}, \emptyset, \text{lab} >$. $\mathcal{N}^i$ will also designate the state of $\mathcal{N}$ after i iterations over the main loop of Algorithm 6, such that $\mathcal{N}^0 = \{N_{root}\}$, and $\mathcal{N}^n$ is the set of nodes of the graph when the algorithm terminates, composed of N_{root} and all its successors (transitively). A similar notation will be used for $\mathcal{E}^i$ and lab^i, such that $\mathcal{G}^i =< \mathcal{N}^i, \mathcal{E}^i, \text{lab}^i >$. Then [Rei87] showed that:

Theorem 8.3.1.1. $\mathcal{D} = \{h(N) \mid N \in \mathcal{N}^n \text{ and } \text{lab}(N) = \emptyset\}$.

Algorithm 6 Reiter's algorithm

```
1: N_root ← CreateNode
2: lab(N_root) ← ComputeConflict(K)
3: if lab(N_root) ≠ ∅ then
4:     EnQueue(F, N_root)
5: end if
6: while IsEmpty(F) = false do
7:     N_1 ← DeQueue(F)
8:     for all φ_1 ∈ lab(N_1) do
9:         if there is a N_2 ∈ 𝒩 s.t. h(N_2) = h(N_1) ∪ {φ} then
10:            SetSuccessor(N_1, N_2, φ_1)
11:        else
12:            if there is no N_3 ∈ 𝒩 s.t. lab(N_3) = ∅ and h(N_3) ⊂ h(N_1) ∪ {φ} then
13:                N_2 ← CreateNode
14:                SetSuccessor(N_1, N_2, φ)
15:                if there is a N_4 ∈ 𝒩 s.t. lab(N_4) ∩ (h(N_1) ∪ {φ}) = ∅ and
   lab(N_4) ≠ ∅ then
16:                    lab(N_2) ← lab(N_4)
17:                    EnQueue(F, N_2)
18:                else
19:                    lab(N_2) ← ComputeConflict(K \ (h(N_1) ∪ {φ}))
20:                    if lab(N_2) ≠ ∅ then
21:                        EnQueue(F, N_2)
22:                        for all N_5 ∈ 𝒩 s.t. lab(N_2) ⊂ lab(N_5) do
23:                            for all φ' ∈ lab(N_5) do
24:                                DeleteEdge(N_5, φ')
25:                                DeleteOrphanNodes(F)
26:                                lab(N_5) ← lab(N_2)
27:                            end for
28:                        end for
29:                    end if
30:                end if
31:            end if
32:        end if
33:    end for
34: end while
```

In other words, after termination, each path from N_{root} to a node labeled with $\emptyset$ is a diagnosis, and conversely.

Another formulation of this algorithm can be found in [Sch05] (but due to place limitation, it was only a partial description). In particular, the authors investigated for DLs the impact on performance of the size of the sets returned by function ComputeConflict(X) (provided $X \cup \Theta \vdash \bot$). Three alternative settings were tested by [Sch05]: either ComputeConflict returns the whole X, or ComputeConflict returns a minimal subset of X inconsistent with Θ (or in other words it returns some $V \in \mathcal{V}$ such that $V \subseteq X$), or ComputeConflict returns some $X' \subseteq X$ inconsistent with Θ, of an intermediate size, obtained with a greedy strategy.[14] The smaller the sets returned by procedure ComputeConflict, the more efficient Reiter's algorithm is. But computing minimal conflicts can be costly, such that a compromise may need to be found. In practice, for the experiments run by [Sch05], using the whole X led to a blowup in the width of the graph, and only the two latter solutions (either minimal or small conflicts) yielded some results. Reiter's algorithm could not be run until termination though, and its depth was bounded by an integer m, which guarantees that all diagnoses of cardinality m or inferior have been computed.

In the case where minimal conflicts are returned by ComputeConflict, the algorithm is actually slightly simplified, because lines 22 to 28 do not need to be executed, or in other words, the graph is never trimmed. This simplified version of Reiter's algorithm was the one used by [FS05, Kal06, QHH+08, Hor11], relying on different black-box or glass-box algorithms which compute one justification (presented

[14] Note that in the two latter cases, a glass-box algorithm can be used to compute either V or X', such that technically, the whole procedure may be considered as a hybrid black-box/glass-box one. Such a glass-box algorithm returns only *one* (possibly minimal) conflict though, and therefore differs from the one presented in the next section, which aims at computing *all* minimal conflicts, not a single one.

in Chapter 3 Section 3.5) for ComputeConflict. Interestingly, this simplified version of Reiter's algorithm was used to obtain (a subset of) $\mathcal{D}$ by [FS05, QHH+08], whereas [Hor11] used it to compute $\mathcal{V}$ instead. This last use of Reiter's algorithm is legitimated by the following observation (proven in section 8.8.3.2):

Proposition 8.3.1.1. $\mathcal{V} = \{\text{lab}^n(N) \mid N \in \mathcal{N}^n\} \setminus \{\emptyset\}\}$

In other words, after termination, each node label in the graph which is not $\emptyset$ is a minimal conflict, and conversely, each minimal conflict is the label of some node in the graph.

But an even tighter correspondence between the computation of $\mathcal{V}$ and the computation of $\mathcal{D}$ with Reiter's algorithm can actually be established. Some additional notation will be useful here. Let $< K, \Theta >$ designate an admissible input of algorithm 6 (for inconsistency), i.e. $K \cup \Theta \vdash \bot$, $\Theta \nvdash \bot$, Θ is the set of axioms to be preserved and K the set of axioms which may be discarded.

It will be shown that given an input $< K_1, \Theta_1 >$, if all diagnoses for $< K_1, \Theta_1 >$ have been computed after i execution steps of algorithm 6, but all minimal conflicts have not, then there is another input $< K_2, \Theta_2 >$ such that applying Reiter's algorithm to $< K_2, \Theta_2 >$ may yield after k execution steps a graph isomorphic to the one produced for $< K_1, \Theta_1 >$, but such that at least one diagnosis for $< K_2, \Theta_2 >$ has not been computed yet. Or intuitively, it cannot be guaranteed that $\mathcal{D}$ has been computed if all elements of $\mathcal{V}$ have not been computed as well.

If $< K, \Theta >$ is an admissible input of algorithm 6, $\mathcal{V}_{<K,\Theta>}$ will designate all minimal conflicts for $< K, \Theta >$, and $\mathcal{D}_{<K,\Theta>}$ all diagnoses for $< K, \Theta >$. In addition, if $\mathcal{G} =< \mathcal{N}, \mathcal{E}, \text{lab} >$, then for each $N \in \mathcal{N}$, $h_{\mathcal{G}}(N)$ will designate $h(N)$ above, but for the graph $\mathcal{G}$.

Finally, if Γ is a finite set of axioms, mapInt(Γ) will designate the set of bijective

mappings from Γ to $\{1, 2, .., |\Gamma|\}$. In addition, if $q \in \text{mapInt}(\Gamma)$, then l_q will be defined by $l_q(X) = \{q(\phi) \mid \phi \in X\}$, i.e. l_q maps a set of axioms to the corresponding set of integers according to q.

Then the following is shown to hold in Section 8.8.3.3:

Proposition 8.3.1.2. Let $\mathcal{G}_1^i =< \mathcal{N}, \mathcal{E}, \text{lab}_1 >$ be a possible state of a graph built during an execution of algorithm 6 for $< K_1, \Theta_1 >$, such that $\mathcal{D}_{<K_1,\Theta_1>} = \{h_{\mathcal{G}_1}(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_1(N) = \emptyset\}$, but there is a $V \in \mathcal{V}_{<K_1,\Theta_1>} \setminus \{\text{lab}_1(N) \mid N \in \mathcal{N}\}$. Then for any $q_1 \in \text{mapInt}(K_1)$, there is a $< K_2, \Theta_2 >$, a $q_2 \in \text{mapInt}(K_2)$ and a possible state $\mathcal{G}_2^k =< \mathcal{N}, \mathcal{E}, \text{lab}_2 >$ of a graph for $< K_2, \Theta_2 >$ such that $< \mathcal{N}, \mathcal{E}, l_{q_1} \circ \text{lab}_1 >=< \mathcal{N}, \mathcal{E}, l_{q_2} \circ \text{lab}_2 >$, but $\mathcal{D}_{<K_2,\Theta_2>} \setminus \{h_{\mathcal{G}_2}(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_2(N) = \emptyset\} \neq \emptyset$.

In other words, at any step of the execution for some input $< K_1, \Theta_1 >$, if all elements of $\mathcal{V}_{<K_1,\Theta_1>}$ have not been computed yet, then it cannot be guaranteed that all elements of $\mathcal{D}_{<K_1,\Theta_1>}$ have (even of this is the case in practice), because there is another input $< K_2, \Theta_2 >$ and a graph $\mathcal{G}_2$ for $< K_2, \Theta_2 >$ such that $\mathcal{G}_2$ is isomorphic to the graph $\mathcal{G}_1$ for $< K_1, \Theta_1 >$, but at least one $D \in \mathcal{D}_{<K_2,\Theta_2>}$ is still missing in $\mathcal{G}_2$.

The converse holds as well. Given an input $< K_1, \Theta_1 >$, if all minimal conflicts for $< K_1, \Theta_1 >$ have been computed after i execution steps of algorithm 6, but all diagnoses have not, then there is another input $< K_2, \Theta_2 >$ such that applying Reiter's algorithm to $< K_2, \Theta_2 >$ may yield after k execution steps a graph isomorphic to the one produced for $< K_1, \Theta_1 >$, but such that some minimal conflicts for $< K_2, \Theta_2 >$ have not been computed yet. Or intuitively, it cannot be guaranteed that $\mathcal{V}$ has been computed if all elements of $\mathcal{D}$ have not been computed as well.

More formally the following is shown to hold in Section 8.8.3.4:

Proposition 8.3.1.3. Let $\mathcal{G}_1^i =< \mathcal{N}, \mathcal{E}, \text{lab}_1 >$ be a possible state of a graph built during an execution of algorithm 6 for $< K_1, \Theta_1 >$, such that $\mathcal{V}_{<K_1,\Theta_1>} = \{\text{lab}_1(N) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}$, but there is a $D \in \mathcal{D}_{<K_1,\Theta_1>} \setminus \{h_{\mathcal{G}_1}(N) \mid N \in \mathcal{N}$ and $\text{lab}_1(N) = \emptyset\}$. Then for any $q_1 \in \text{mapInt}(K_1)$, there is a $< K_2, \Theta_2 >$, a $q_2 \in \text{mapInt}(K_2)$ and a possible state $\mathcal{G}_2^k =< \mathcal{N}, \mathcal{E}, \text{lab}_2 >$ of a graph for $< K_2, \Theta_2 >$ such that $< \mathcal{N}, \mathcal{E}, l_{q_1} \circ \text{lab}_1 >=< \mathcal{N}, \mathcal{E}, l_{q_2} \circ \text{lab}_2 >$, but $\mathcal{V}_{<K_2,\Theta_2>} \setminus \{\text{lab}_2(N) \mid N \in \mathcal{N}\} \neq \emptyset$.

In other words, at any step of the execution for some input $< K_1, \Theta_1 >$, if all elements of $\mathcal{D}_{<K_1,\Theta_1>}$ have not been computed yet, then it cannot be guaranteed that all elements of $\mathcal{V}_{<K_1,\Theta_1>}$ have (even of this is the case in practice), because there is another input $< K_2, \Theta_2 >$ and a graph $\mathcal{G}_2$ for $< K_2, \Theta_2 >$ such that $\mathcal{G}_2$ is isomorphic to the graph $\mathcal{G}_1$ for $< K_1, \Theta_1 >$, but at least one $V \in \mathcal{V}_{<K_2,\Theta_2>}$ is still missing in $\mathcal{G}_2$.

This second observation is interesting for the problem of concern in this Chapter, which is computing $\mathcal{D}$ (or equivalently $\mathcal{R}_\subseteq$) or a selection of elements of $\mathcal{D}$ for some $< K, \Theta >$. It shows that the strategy which consists in computing $\mathcal{V}$ with Reiter's algorithm before computing $\mathcal{D} = \min_\subseteq(\text{hs}(\mathcal{V}))$ from $\mathcal{V}$ is indeed inefficient, because $\mathcal{D}$ has already been computed during the execution of Reiter's algorithm to obtain $\mathcal{V}$.

8.3.2 Glass-box: saturated tableau and axiom pinpointing

This section discusses glass-box techniques used in the DL literature in order to compute $\mathcal{V}$ or $\mathcal{D}$, and in particular the fact that even if both problems are equally hard as far as worst-case complexity is concerned, for a same input, computing one of $\mathcal{D}$ or $\mathcal{V}$ may be more costly in practice than computing the other. This contrasts with Reiter's algorithm, described in the previous section, for which the cost of both operations is identical.

Again, the focus is put on the case of inconsistency for the sake of readability. In the rest of this section, A or A_i will designate an atomic DL concept, R or R_i an atomic DL role, e or e_i an individual, and C or C_i a (possibly concept) DL concept.

Glass-box techniques for KB debugging in DLs are prototypically based on a tableau algorithm, introduced in Chapter 2 Section 2.3.8.2, and more exactly on a saturated tableau with a tracing mechanism, introduced in Chapter 3 Section 3.5.1.2, which allows here for keeping track of the axioms involved in the inconsistency. A relatively traditional (and slightly simplified[15]) execution of a saturated tableau with tracing is presented here, in that it is assumed that the saturated tableau is executed until termination (i.e. until no expansion rule can be applied anymore), before either $\mathcal{V}$ or $\mathcal{D}$ is computed out of the "traces" produced by the tracing mechanism. Alternatively, [MLBP06] propose to compute $\mathcal{D}$ during the execution of the tableau, but their procedure nonetheless runs the saturated tableau until termination, such that this has no incidence on the cost of the operation.[16] In other words, for the specific point discussed here, describing the computation of $\mathcal{D}$ with the algorithm described in [MLBP06] or with a more conventional saturated tableau is only a matter of presentation.

So it is assumed here that the saturated tableau is executed until termination, and the discussion will focus on the subsequent operation, which is the computation of either $\mathcal{V}$ or $\mathcal{D}$ after termination of the tableau.

But before this, a more detailed description of the execution of a saturated tableau may be useful. As a reminder, a tableau algorithm for the consistency of $K \cup \Theta$

[15] see the omission of blocking conditions below

[16] An detailed proof of this claim may be tedious. The equivalence between the two procedures can be seen from the fact that the incisions produced during the execution of the algorithm described in [MLBP06] are not necessarily minimal, and more specifically that the whole family $\mathcal{F}$ of incisions described below may need to be computed during the execution of both algorithms.

extends a family $\mathcal{W}$ of sets of DL formulas, and the successive states of $\mathcal{W}$ will once again be designated with $\mathcal{W}_0, .., \mathcal{W}_n$. Before the execution, each TBox axiom in $K \cup \Theta$ of the form $C_1 \sqsubseteq C_2$ is normalized syntactically into an axiom of the form $A \sqsubseteq C$, where A is an atomic DL concept and C is in NNF. Then $\mathcal{W}$ is initialized with $\mathcal{W}_0 = \{W_0\}$, such that $W_0 = K \cup \Theta \cup \{\top(e) \mid e \in N_{Ind}(K \cup \Theta)\}$ if $N_{Ind}(K \cup \Theta) \neq \emptyset$, or $W_0 = K \cup \Theta \cup \{\top(e')\}$ if $N_{Ind}(K \cup \Theta) = \emptyset$, with e' a fresh individual. $\mathcal{W}$ is extended by iterative applications of a set of expansion rules to each $W \in \mathcal{W}$. If $W \in \mathcal{W}_i$ for some $i \in \{0, .., n\}$, and if an expansion rule is applied to W, then $\mathcal{W}_i$ becomes $\mathcal{W}_{i+1} = (\mathcal{W}_i \setminus \{W\}) \cup \text{expans}(W)$, where expans$(W)$ is a family of (one or two) sets of DL formulas (an example of execution will be given below). According to the notation introduced in Chapter 2 Section 2.3.8.2, expans$^*(W)$ will designate W and its (immediate and transitive) expansions.

For debugging, a tableau algorithm is extended with a tracing mechanism (see Chapter 3 Section 3.5.1.2), which keeps track for each $W \in \mathcal{W}$ of the axioms involved in the derivation of each formula $\gamma \in W$. In addition, if all elements of either $\mathcal{V}$ or $\mathcal{D}$ are to be computed, expansion rules are applied until saturation. For readability, the choice is made here to represent the tracing mechanism by sets of labels,[17] such that the elements of each $W \in \mathcal{W}$ are not DL formulas as in a standard tableau, but pairs $< \gamma, \Lambda >$, with γ a DL formula, and $\Lambda \subseteq K \cup \Theta$ a set of axioms involved in the derivation of γ.

A basic example of execution in the DL $\mathcal{ALC}$ with individuals (i.e. with an ABox) will be fully developed. In order to keep the example simple, it is assumed that $\Theta = \emptyset$, such that the (inconsistent) input KB is $K \cup \Theta = K$, which also implies that $\mathcal{V} = \mathcal{J}$, as explained in Section 8.1.3.5 above. The input set of axioms K is

[17] rather than by boolean formulas attributed to derived DL formulas, like in [BP10], but this is only a matter of taste, both notations being equivalent.

given by example 8.3.1:

Ex 8.3.1. $K = \{$

(1) $A_1 \sqsubseteq A_2$,

(2) $A_1 \sqsubseteq \neg A_2 \sqcup \forall R_2.A_3$,

(3) $A_2 \sqsubseteq \exists R_1.A_3$,

(4) $A_1 \sqsubseteq \forall R_1.A_4$,

(5) $A_3 \sqsubseteq \neg A_4$,

(6) $A_2 \sqsubseteq \forall R_2.A_4$,

(7) $A_1(e_1)$,

(8) $A_3(e_2)$,

(9) $R_2(e_1, e_2)$

For the sake of brevity, the axioms of K will be designated with their index, for instance $A_1 \sqsubseteq A_2$ with 1, $A_1 \sqsubseteq \neg A_2 \sqcup \forall R_2.A_3$ with 2, etc. Each of these axioms is already normalized syntactically.

The set of expansion rules for $\mathcal{ALC}$ with individuals is reproduced in figure 8-1 for the sake of the example. The formulation of the $\exists$ rule is on purpose incomplete, in order to keep the presentation simple. In practice, additional *blocking* conditions must be implemented (see [BCM+03]) to deal with the case where the TBox of the input KB contains cycles and existential quantifiers, for instance if $\{A \sqsubseteq \exists R.B$, $B \sqsubseteq A$, $A(e)\} \subseteq K$. But K in example 8.3.1 is a so-called *acyclic* TBox, which guarantees that the saturated tableau terminates here without the need for a blocking mechanism. The order in which these rules are applied may also have an impact on the performance of the tableau algorithm. But this is not relevant for this discussion either, which focuses on the cost of the computation of $\mathcal{V}$ and $\mathcal{D}$ after termination of the saturated tableau.

$\sqsubseteq$ If $\{< A \sqsubseteq C, \Lambda_1 >, < A(e), \Lambda_2 >\} \subseteq W$ and $< C(e), \Lambda_1 \cup \Lambda_2 > \notin W$,
then $\text{expans}(W) = \{W \cup \{< C(e), \Lambda_1 \cup \Lambda_2 >\}\}$

$\sqcap$ If $< C_1 \sqcap C_2(e), \Lambda > \in W$ and $(< C_1(e), \Lambda > \notin W$ or $< C_2(e), \Lambda > \notin W)$,
then $\text{expans}(W) = \{W \cup \{< C_1(e), \Lambda >, < C_2(e), \Lambda >\}\}$

$\sqcup$ If $< C_1 \sqcup C_2(e), \Lambda > \in W$ and $< C_1(e), \Lambda > \notin W$ and $< C_2(e), \Lambda > \notin W$,
then $\text{expans}(W) = \{W \cup \{< C_1(e), \Lambda >\}, W \cup \{< C_2(e), \Lambda >\}\}$

$\exists$ If $< \exists R.C(e_1), \Lambda > \in W$ and there is no e' such that $< R(e_1, e'), \Lambda > \in W$,
then $\text{expans}(W) = \{W \cup \{< R(e_1, e_2), \Lambda >, < C(e_2), \Lambda >, < \top(e_2), \emptyset >\}\}$,
with $e_2 \notin N_{Ind}(W)$

$\forall$ If $\{< \forall R.C(e_1), \Lambda_1 >, < R(e_1, e_2), \Lambda_2 >\} \subseteq W$ and $< C(e_2), \Lambda_1 \cup \Lambda_2 > \notin W$,
then $\text{expans}(W) = \{W \cup \{< C(e_2), \Lambda_1 \cup \Lambda_2 >\}\}$

Figure 8-1: expansion rules for a tableau algorithm with tracing in $\mathcal{ALC}$ with individuals (blocking conditions omitted)

Here is a possible execution of a saturated tableau for the consistency of K in example 8.3.1:

(a) $\mathcal{W} \leftarrow \{W_0\}$

(b:$\sqsubseteq$) $W_1 = W_0 \cup \{< A_2(e_1), \{1, 7\} >\}$ $\mathcal{W} \leftarrow \{W_1\}$

(c:$\sqsubseteq$) $W_2 = W_1 \cup \{< \exists R_1.A_3(e_1), \{1, 3, 7\} >\}$ $\mathcal{W} \leftarrow \{W_2\}$

(d:$\exists$) $W_3 = W_2 \cup \{< R_1(e_1, e_3), \{1, 3, 7\} >, < A_3(e_3), \{1, 3, 7\} >, < \top(e_3), \emptyset\} >\}$
$\mathcal{W} \leftarrow \{W_3\}$

(e:$\sqsubseteq$) $W_4 = W_3 \cup \{< \forall R_1.A_4(e_1), \{4, 7\} >\}$ $\mathcal{W} \leftarrow \{W_4\}$

(f:$\forall$) $W_5 = W_4 \cup \{< A_4(e_3), \{1, 3, 4, 7\} >\}$ $\mathcal{W} \leftarrow \{W_5\}$

(g:$\sqsubseteq$) $W_6 = W_5 \cup \{< \neg A_4(e_3), \{1, 3, 5, 7\} >\}$ $\mathcal{W} \leftarrow \{W_6\}$

(h:$\sqsubseteq$) $W_7 = W_6 \cup \{< \forall R_2.A_4(e_1), \{1, 6, 7\} >\}$ $\mathcal{W} \leftarrow \{W_7\}$

(i:$\forall$) $W_8 = W_7 \cup \{< A_4(e_2), \{1, 6, 7, 9\} >\}$ $\mathcal{W} \leftarrow \{W_8\}$

(j:$\sqsubseteq$) $W_9 = W_8 \cup \{< \neg A_4(e_2), \{5, 8\} >\}\}$ $\mathcal{W} \leftarrow \{W_9\}$

(k:$\sqsubseteq$) $W_{10} = W_9 \cup \{< \neg A_2 \sqcup \forall R_2.A_3(e_1), \{2, 7\} >\}$ $\mathcal{W} \leftarrow \{W_{10}\}$

(l:$\sqcup$) $W_{11} = W_{10} \cup \{< \neg A_2(e_1), \{2,7\} >\}$,

$W_{12} = W_{10} \cup \{< \forall R_2.A_3(e_1), \{2,7\} >\}$ $\mathcal{W} \leftarrow \{W_{11}, W_{12}\}$

(m:$\forall$) $W_{13} = W_{12} \cup \{< A_3(e_2), \{2,7,9\} >\}$ $\mathcal{W} \leftarrow \{W_{11}, W_{13}\}$

(n:$\sqsubseteq$) $W_{14} = W_{13} \cup \{< \neg A_4(e_2), \{2,5,7,9\} >\}$ $\mathcal{W} \leftarrow \{W_{11}, W_{14}\}$

After termination, $\mathcal{W}_n = \{W_{11}, W_{13}\}$. This example illustrates several properties of the algorithm. First, the size of $\mathcal{W}$ may increase only after application of the $\sqcup$ rule (for instance after step l).

A second observation is that derived formulas are not all consequences of K. This is obvious for an application of the $\exists$ rule, because some fresh individuals are introduced. For instance, after execution step d, the formula $A_3(e_3)$ has been derived, but because $e_3 \notin \text{sig}(K)$ and $K \not\models \top \sqsubseteq A_3$, $K \not\models A_3(e_3)$. But this may also hold of formulas whose signature is a subset of $\text{sig}(K)$, after an application of the $\sqcup$ rule. Let $v(W)$ designate the formulas of W, omitting their labels, i.e. $\gamma \in v(W)$ iff $< \gamma, \Lambda >\in W$ for some $\Lambda \subseteq K$. And for any set of formulas Ω, let $\Omega_{|\,\text{sig}(K)} = \{\omega \in \Omega \mid \text{sig}(\{\omega\}) \subseteq \text{sig}(K)\}$. Then it does not hold in general that for all $W \in \mathcal{W}$, $v(W)_{|\,\text{sig}(K)} \subseteq \text{Cn}(K)$. For instance, after step l, neither $\neg A_2(e_1)$ nor $\forall R_2.A_3(e_1)$ is a consequence of K, but $\neg A_2 \sqcup \forall R_2.A_3(e_1)$ is. Instead, the tableau procedure guarantees at each execution step that $\bigcap_{W \in \mathcal{W}} \text{Cn}(v(W))_{|\,\text{sig}(K)} = \text{Cn}(K)$. Another way to view this is that the two W', W'' generated by an application of the $\sqcup$ rule are two alternative (attempted) models under construction for K.

A third observation is that if $\gamma \in v(W)$ for some $W \in \mathcal{W}$ after termination, then γ may have been derived multiple times from different sets of axioms, i.e. there may be $< \gamma, \Lambda_1 >, < \gamma, \Lambda_2 >\in W$ such that $\Lambda_1 \neq \Lambda_2$. Furthermore, it may be the case that neither $\Lambda_1 \not\subseteq \Lambda_2$ nor $\Lambda_2 \not\subseteq \Lambda_1$ holds, such that two alternative explanations can

intuitively be found for $\gamma \in v(W)$. For instance, $A_3(e_2) \in v(W_{13})$, but it appears twice in W_{13}, namely in $< A_3(e_2), \{8\} >$ and in $< A_3(e_2), \{2,7,9\} >$, such that $\{8\}$ and $\{2,7,9\}$ are two alternative explanations for $A_3(e_2)$ in W_{13}. Similarly, both $< \neg A_4(e_2), \{5,8\} >$ and $< \neg A_4(e_2), \{2,5,7,9\} >$ are elements of W_{14}.

As a reminder of Chapter 2 Section 2.3.8.2, the clashes under consideration here are of the form $\{< \bot(e), \Lambda >\}$, with e an individual, or $\{< A(e), \Lambda_1 >, < \neg A(e), \Lambda_2 > \}$, with A an atomic concept and e an individual. Now let clash(W) designate all clashes in some $W \in \mathcal{W}$, and for any clash $M \in$ clash(W), let $r(M) = \bigcup_{<\gamma,\Lambda> \in r(M)} \Lambda$. In example 8.3.1, after termination, there are three clashes in W_{11}, i.e. clash$(W_{11}) = \{M_1^{W_{11}}, M_2^{W_{11}}, M_3^{W_{11}}\}$, with:

$$M_1^{W_{11}} = \{< A_4(e_2), \{1,6,7,9\} >, < \neg A_4(e_2), \{5,8\} >\}$$
$$M_2^{W_{11}} = \{< A_4(e_3), \{1,3,4,7\}, < \neg A_4(e_3), \{1,3,5,7\} >\}$$
$$M_3^{W_{11}} = \{< A_2(e_1), \{1,7\} >, < \neg A_2(e_1), \{2,7\} >\}$$

So $r(M_1^{W_{11}}) = \{1,6,7,9\} \cup \{5,8\} = \{1,5,6,7,8,9\}$, $r(M_2^{W_{11}}) = \{1,3,4,7\} \cup \{1,3,5,7\} = \{1,3,4,5,7\}$, and $r(M_3^{W_{11}}) = \{1,7\} \cup \{2,7\} = \{1,2,7\}$. In W_{14}, there are three clashes too, namely:

$$M_1^{W_{14}} = \{< A_4(e_2), \{1,6,7,9\} >, < \neg A_4(e_2), \{5,8\} >\}$$
$$M_2^{W_{14}} = \{< A_4(e_3), \{1,3,4,7\}, < \neg A_4(e_3), \{1,3,5,7\} >\}$$
$$M_3^{W_{14}} = \{< A_4(e_2), \{1,6,7,9\}, < \neg A_4(e_2), \{2,5,7,9\} >\}$$

Similarly, $r(M_1^{W_{14}}) = \{1,5,6,7,8,9\}$, $r(M_2^{W_{14}})\{1,3,4,5,7\}$, and $r(M_3^{W_{14}}) = \{1,2,5,6,7,9\}$.

As mentioned in Chapter 3 Section 3.5.1.2, after termination of the saturated tableau, if $\mathcal{W}_n = \{W_1, .., W_m\}$, then the following holds:[18]

[18] A complete proof will not be provided here, and the reader is referred to [SC03, MLBP06, BP10] instead.

Theorem 8.3.2.1. $\mathcal{V} = \min_{\subseteq}\{\bigcup_{k=1}^{m}\{r(M) \mid < M_1, .., M_m > \in \prod_{W_k \in \mathcal{W}_n} \text{clash}(W_k)\}$

In order to simplify notation, let $\mathcal{H} = \{\bigcup_{k=1}^{m}\{r(M) \mid < M_1, .., M_m > \in \prod_{W_k \in \mathcal{W}_n} \text{clash}(W_k)\}$, such that from the above theorem, $\mathcal{V} = \min_{\subseteq} \mathcal{H}$. Intuitively, if one clash M_k is selected in each $W_k \in \mathcal{W}_n$, then the union of all axioms of K involved in each of these clashes is a conflict $H \in \mathcal{H}$, and the procedure guarantees that $\mathcal{V} \subseteq \mathcal{H}$. Then because the elements of $\mathcal{V}$ are minimal conflicts, $\mathcal{V} = \min_{\subseteq} \mathcal{H}$ must hold.

Another way to represent this is the boolean *pinpointing formula* $\alpha(\mathcal{W}_n)$ used in [BP10], provided a bijective mapping ρ from K to a set of propositional variables. It is defined (up to equivalence) as:

Definition 8.3.2.1. $\alpha(\mathcal{W}_n) = \bigwedge_{W \in \mathcal{W}_n} \bigvee_{M \in \text{clash}(W)} \bigwedge_{\phi \in r(M)} \rho(\phi)$

In example 8.3.1, Let $\rho(A_1 \sqsubseteq A_2) = p_1$, $\rho(A_1 \sqsubseteq \neg A_2 \sqcup \forall R_2.A_3) = p_2$, etc.

Then the pinpointing formula is:

$$\alpha(\mathcal{W}) = \Big((p_1 \wedge p_5 \wedge p_6 \wedge p_7 \wedge p_8 \wedge p_9) \vee (p_1 \wedge p_3 \wedge p_4 \wedge p_5 \wedge p_7) \vee (p_1 \wedge p_2 \wedge p_7)\Big) \wedge$$
$$\Big((p_1 \wedge p_5 \wedge p_6 \wedge p_7 \wedge p_8 \wedge p_9) \vee (p_1 \wedge p_3 \wedge p_4 \wedge p_5 \wedge p_7) \vee (p_1 \wedge p_2 \wedge p_5 \wedge p_6 \wedge p_7 \wedge p_9)\Big)$$

Another way to formulate the relation between $\mathcal{V}$ and $\mathcal{W}$ is that $V \in \mathcal{V}$ iff $\{\rho(\phi) \mid \phi \in V\}$ is a prime implicant of $\alpha(\mathcal{W}_n)$, i.e. a minimal valuation satisfying $\alpha(\mathcal{W}_n)$. In this example, the prime implicants of $\alpha(\mathcal{W})$ are $\{p_1, p_5, p_6, p_7, p_8, p_9\}$, $\{p_1, p_3, p_4, p_5, p_7\}$ and $\{p_1, p_2, p_5, p_6, p_7, p_9\}$, such that $\mathcal{V} = \{\{1, 5, 6, 7, 8, 9\}, \{1, 3, 4, 5, 7\}, \{1, 2, 5, 6, 7, 9\}\}$.

Now for each $W \in \mathcal{W}_n$, let $s(W)$ be the family of sets of axioms defined by $s(W) = \{r(M) \mid M \in \text{clash}(W)\}$. For instance, $s(W_{10}) = \{\{1, 5, 6, 7, 8, 9\}, \{1, 3, 4, 5, 7\}, \{1, 2, 7\}\}$. Then the following holds:[19]

[19] Again, a complete proof will not be provided here, and the reader is referred to [MLBP06, BP10] instead.

Theorem 8.3.2.2. $\mathcal{D} = \min_{\subseteq}\{\text{hs}(s(W)) \mid W \in \mathcal{W}_n\}$

Intuitively, each $\text{hs}(s(W))$ for some $W \in \mathcal{W}_n$ is also an incision for K. For instance, $\{1\}$ and $\{2,4,6\}$ are two (minimal) hitting sets for $s(W_{10})$, and therefore both are incisions for K. $\{1\}$ is not only an incision, but also a diagnosis for K, i.e. it is minimal wrt $\subseteq$ among all incisions for K. But $\{2,4,6\}$, even if it is a minimal hitting set for $s(W_{10})$, is not a diagnosis for K, because there is a hitting set for $s(W_{13})$ which is a strict subset of $\{2,4,6\}$, namely $\{4,6\}$.

The procedure also guarantees that $\mathcal{D} \subseteq \{\text{hs}(s(W)) \mid W \in \mathcal{W}_n\}$, so because the elements of $\mathcal{D}$ are minimal incisions, $\mathcal{D} = \min_{\subseteq}\{\text{hs}(s(W)) \mid W \in \mathcal{W}_n\}$ holds.

Equivalently, each $D \in \mathcal{D}$ corresponds to a prime implicate of $\alpha(\mathcal{W})$, i.e. $D \in \mathcal{D}$ iff $\{\rho(\phi) \mid \phi \in K\} \backslash \{\rho(\phi) \mid \phi \in D\}$ is a maximal valuation (over $\{\rho(\phi) \mid \phi \in K\}$) verifying $\alpha(\mathcal{W}_n)$. In example 8.3.1, the prime implicates of $\alpha(\mathcal{W})$ are $\{p_1\}, \{p_5\}, \{p_7\}, \{p_3, p_6\}$, $\{p_4, p_6\}, \{p_3, p_9\}, \{p_4, p_9\}, \{p_2, p_3, p_8\}, \{p_2, p_4, p_8\}$, such that $\mathcal{D} = \{\{1\}, \{5\}, \{7\}, \{3,6\}$, $\{4,6\}, \{3,9\}, \{4,9\}, \{2,3,8\}, \{2,4,8\}\}$.

If β is an arbitrary boolean formula, computing its primes implicants or its prime implicates are both as hard in the worst case. But the reasons for a potential blowup in the cost of each operation differ, such that for a same input, computing $\mathcal{V}$ or $\mathcal{D}$ may in practice have different costs, and understanding the respective sources of intractability in each case may provide a useful insight, in order to determine in which cases computing $\mathcal{V}$ or computing $\mathcal{D}$ is a more realistic scenario for a specific input.

For the computation of $\mathcal{D}$, in the worst case, the following may hold:

- for each $W \in \mathcal{W}_n$, $\min_{\subseteq} s(W) = s(W)$
- for each $W_1, W_2 \in \mathcal{W}_n$ such that $W_1 \neq W_2$, for all $F_1 \in \min_{\subseteq} \text{hs}(s(W_1))$, for all $F_2 \in \min_{\subseteq} \text{hs}(s(W_2))$, $F_1 \not\subseteq F_2$.

In this specific case, from theorem 8.3.2.2, if $\mathcal{F} = \{\min_{\subseteq} \mathrm{hs}(s(W)) \mid W \in \mathcal{W}_n\}$, then $\mathcal{D} = \mathcal{F}$ must hold. In other words, in the worst case, all minimal hitting sets for each $s(W)$ such that $W \in \mathcal{W}_n$ may need to be computed.

For the computation of $\mathcal{V}$, let $\mathcal{S} = \{s(W) \mid W \in \mathcal{W}_n\}$. Then an alternative definition of $\mathcal{H}$ above is $\mathcal{H} = \{\bigcup \mathcal{Z} \mid \mathcal{Z} \in \min \mathrm{hs}(\mathcal{S})\}$.[20] In the worst case, the following may hold:

- for each $W \in \mathcal{W}_n$, $\min_{\subseteq} s(W) = s(W)$[21]
- for all $\mathcal{Z}_1, \mathcal{Z}_2 \in \min_{\subseteq} \mathrm{hs}(\mathcal{S})$ such that $\mathcal{Z}_1 \neq \mathcal{Z}_2$, $\bigcup \mathcal{Z}_1 \not\subseteq \bigcup \mathcal{Z}_2$

In this specific case, from theorem 8.3.2.1, $\mathcal{V} = \mathcal{H}$, so $\mathcal{H}$ needs to be computed. Then because $|\mathcal{H}| = |\min_{\subseteq} \mathrm{hs}(\mathcal{S})|$ also holds in this case, and because $\mathcal{H} = \{\bigcup \mathcal{Z} \mid \mathcal{Z} \in \min_{\subseteq} \mathrm{hs}(\mathcal{S})\}$, $\min_{\subseteq} \mathrm{hs}(\mathcal{S})$ needs to be computed as well.

Now let a be the size of the largest set of axioms associated to any clash in any $W \in \mathcal{W}_n$, i.e. $a = \max_{W \in \mathcal{W}_n, M \in \mathrm{clash}(W)} |r(M)|$. Then let b be the largest number of clashes in some $W \in \mathcal{W}_n$, i.e. $b = \max_{W \in \mathcal{W}_n} |\mathrm{clash}(W)|$, and let $c = |\mathcal{W}_n|$.

In the abovementioned worst case for $\mathcal{D}$, $\mathcal{D} = \mathcal{F}$, i.e. $\mathcal{D}$ is the collection of all minimal hitting sets for all $s(W)$ such that $W \in \mathcal{W}_n$. But there can be at most $c \cdot a^b$ of them, such that computing $\mathcal{D}$ after termination of a saturated tableau (for instance with a brute-force enumeration of all $\mathrm{hs}(s(W))$, followed by a selection of the minimal elements wrt $\subseteq$) is at most linear in c, at most polynomial in a and at most exponential in b. On the other hand, in this worst case for $\mathcal{D}$ still, and if

[20] $\mathcal{S}$ is a family of families of sets of axioms, so each $\mathcal{Z} \in \min_{\subseteq} \mathrm{hs}(\mathcal{S})$ is a family of sets of axioms, and not simply a set of axioms.

[21] The reason for this first precaution may not be obvious at first sight. If $\mathcal{S}' = \{\min_{\subseteq} s(W) \mid W \in \mathcal{W}_n\}$, then for each $\mathcal{Z} \in \min_{\subseteq} \mathrm{hs}(\mathcal{S})$, there is a $\mathcal{Z}' \in \min_{\subseteq} \mathrm{hs}(\mathcal{S}')$ such that $\bigcup \mathcal{Z}' \subseteq \bigcup \mathcal{Z}$, and therefore $\min_{\subseteq}\{\bigcup \mathcal{Z}' \mid \mathcal{Z}' \in \min_{\subseteq} \mathrm{hs}(\mathcal{S}')\} = \min_{\subseteq}\{\bigcup \mathcal{Z} \mid \mathcal{Z} \in \min_{\subseteq} \mathrm{hs}(\mathcal{S})\}$. So if for some $W \in \mathcal{W}_n$, $\min_{\subseteq} s(W) \subset s(W)$ holds, the cost of computing $\mathcal{E}$ may be lowered by reducing each $s(W)$ to $\min_{\subseteq} s(W)$ beforehand, which can be done in time polynomial in $\sum_{W \in \mathcal{W}_n} \sum_{M \in \mathrm{clash}(W)} |r(M)|$.

$b > 1$, there may be a $W \in \mathcal{W}_n$ such that $|s(W)| = b$ and for all $M \in \text{clash}(W)$, $|r(M)| > 1$. Therefore the cost of the computation of $\mathcal{D}$ may be exponential in b.

In the abovementioned worst case for $\mathcal{V}$, $\mathcal{V} = \mathcal{H}$, where each $H \in \mathcal{H}$ is the union of all elements of a minimal hitting set for $\mathcal{S}$. From the definition of $\mathcal{S}$, $|\mathcal{S}| \leq |\mathcal{W}_n|$, so $|\mathcal{S}| \leq c$, and therefore each $\mathcal{Z} \in \min_{\subseteq} \text{hs}(\mathcal{S})$ is such that $|\mathcal{Z}| \leq c$ as well. Then each $Z \in \mathcal{Z}$ is such that $Z = r(M)$ for some $W \in \mathcal{W}_n$ and some $M \in \text{clash}(W)$. And because $|r(M)| \leq a$, $|Z| \leq a$ must hold. So $|\bigcup \mathcal{Z}| \leq a \cdot c$, such that computing $\bigcup \mathcal{Z}$ out of $\mathcal{Z}$ is in $O(ac \log ac)$. Then there are at most b^c elements in $\min_{\subseteq} \text{hs}(\mathcal{S})$, such that computing $\mathcal{V} = \mathcal{H} = \{\bigcup \mathcal{Z} \mid \mathcal{Z} \in \min_{\subseteq} \text{hs}(\mathcal{S})\}$ after termination of a saturated tableau is at most linear in a, at most polynomial in b, and at most exponential in c. On the other hand, in this worst case for $\mathcal{V}$ still, and if $c > 1$, it may be the case that $|\mathcal{W}_n| = c$, and that for all $W \in \mathcal{W}_n$, $|\text{clash}(W)| > 1$. Therefore the computation of $\mathcal{V}$ may be exponential in c.

Finally, in most DLs, b and c are not bounded by each other. As an illustration, for an arbitrarily large integer k, it is possible to find an input KB K such that $b = 1$ and $c = k$. For instance, if $k = 3$, set $K = \{\top \sqsubseteq A_1 \sqcup A_2 \sqcup A_3, \top \sqsubseteq \neg A_1, \top \sqsubseteq \neg A_2, \top \sqsubseteq \neg A_3\}$, with A_1, A_1 and A_3 atomic DL concepts. And conversely, it possible to find an input KB K such that $b = k$ and $c = 1$. For instance, if $k = 3$, set $K = \{\top \sqsubseteq A_1, \top \sqsubseteq \neg A_1, \top \sqsubseteq A_2, \top \sqsubseteq \neg A_2, \top \sqsubseteq A_3, \top \sqsubseteq \neg A_3\}$, with A_1, A_1 and A_3 atomic DL concepts.

So after termination of a saturated tableau, a potential explosion in the cost of the computation of $\mathcal{D}$ may only come from the number of clashes in some $W \in \mathcal{W}_n$, whereas for $\mathcal{V}$, it may only come from the cardinality of $\mathcal{W}_n$, both values being non correlated. Therefore, as opposed to Reiter's algorithm, computing $\mathcal{V}$ or $\mathcal{D}$ for a same input with a saturated tableau may actually have very different empirical computational costs. An interesting continuation would be a more practical investigation

of the type of inputs for which $\mathcal{V}$ or $\mathcal{D}$ may be easier to compute.

Some incorrect simplifications should be avoided here though. In particular, in many DLs, the cardinality of $|\mathcal{W}_n|$ (i.e. c above) is bounded by the number of applications of the $\sqcup$ expansion rule in figure 8-1, triggered by the presence in some $W \in \mathcal{W}$ of the DL concept disjunction operator $\sqcup$. But assuming that $|\mathcal{W}_n|$ is bounded by (or for instance linear in) the number of syntactic occurrences of $\sqcup$ in K is incorrect, because of the syntactic normalization of K before the execution of the tableau, which may introduce additional occurrences of $\sqcup$, but also because a same (normalized) axiom with an occurrence of $\sqcup$ may cause the $\sqcup$ rule to be triggered multiple times during the execution.

8.4 Computing $\mathcal{R}_{\preceq_r}$: algorithm

Given a preference relation $\preceq_a$ over $\bigcup\mathcal{V}$, and assuming $\mathcal{V}$ is known, but not necessarily $\mathcal{R}_\subseteq$, this section provides an algorithm which performs prioritized revision in the sense of definition 8.2.2.3, i.e. which yields the set $\mathcal{R}_{\preceq_r} \subseteq \mathcal{R}_\subseteq$, without the need to compute the whole remainder set $\mathcal{R}_\subseteq$ (unless $\mathcal{R}_{\preceq_r} = \mathcal{R}_\subseteq$).

A first simple observation is that in practice $\preceq_a$ only needs to be defined over $\bigcup\mathcal{V}$. From proposition 8.2.2.2 and lemma 8.1.3.1, each $R \in \mathcal{R}_{\preceq_r}$ is the complement in K of a diagnosis, such that each discarded axiom must be an element of $\bigcup\mathcal{D}$. And from theorem 8.1.3.1, $\bigcup\mathcal{D} \subseteq \bigcup\mathcal{V}$, so each discarded axiom must be an element of $\bigcup\mathcal{V}$. Therefore, even if $\preceq_a$ is defined over $\bigcup\mathcal{V}$ only, it can be extended "for free" to $K \cup \Theta$ by setting all axioms of $K \cup \Theta \setminus (\bigcup\mathcal{V})$ to be maximal wrt $\preceq_a$, and strictly preferred to any axiom of $\bigcup\mathcal{V}$, or in other words, if $(\bigcup\mathcal{V})^{\preceq_a} = (\Delta_1, .., \Delta_n)$, then $(K \cup \Theta)^{\preceq_a} = (\Delta_1, .., \Delta_n, K \cup \Theta \setminus (\bigcup\mathcal{V}))$.

The procedure itself is just an improvement over the computation of $\mathcal{R}_\subseteq$ out

of $\mathcal{V}$, taking advantage of $\preceq_a$. As explained in section 8.1.3.5, obtaining $\mathcal{R}_\subseteq$ (or equivalently $\mathcal{D}$) out of $\mathcal{V}$ amounts to computing $\min_\subseteq \text{hs}(\mathcal{V})$, i.e. all minimal hitting sets for $\mathcal{V}$, which is a well studied problem.[22] It is therefore assumed that some procedure to solve this problem is available (prototypically a breadth-first search tree), which for any finite family of finite sets $\mathcal{X}$ returns $\min_\subseteq \text{hs}(\mathcal{X})$.

Some definitions will be useful. Let $\text{hit} : 2^{2^\mathcal{L}} \times \mathcal{L}^2 \mapsto 2^{2^\mathcal{L}}$ be defined by:

Definition 8.4.0.2. $\text{hit}(\mathcal{X}, Y) = \{X \in \mathcal{X} \mid X \cap Y \neq \emptyset\}$

If $\mathcal{X} \subseteq 2^\mathcal{L}$ and $\preceq_x$ is a total preorder over $\bigcup \mathcal{X}$, let $n = |(\bigcup \mathcal{X})/\sim_x|$. Then the function $\text{pHit} : 2^{2^\mathcal{L}} \times \mathcal{L}^2 \times \{1, 2, .., n\} \mapsto 2^{2^\mathcal{L}}$ is defined by:

Definition 8.4.0.3. $\text{pHit}(\mathcal{X}, \preceq_x, i) = \{X \in \mathcal{X} \mid X \in \text{hit}(\mathcal{X}, (\bigcup \mathcal{X})_i^{\preceq_x})$ and for $1 \leq j < i$, $X \notin \text{hit}(\mathcal{X}, (\bigcup \mathcal{X})_j^{\preceq})\}$

In particular, if $(\bigcup \mathcal{V})^{\preceq_a} = ((\bigcup \mathcal{V})_1^{\preceq_a}, .., (\bigcup \mathcal{V})_n^{\preceq_a})$, then $\text{pHit}(\mathcal{V}, \preceq_a, i)$ returns the elements of $\mathcal{V}$ which are hit by $(\bigcup \mathcal{V})_i^{\preceq_a}$, but not by any lower-ranked equivalence class of $(\bigcup \mathcal{V})^{\preceq_a}$.

If $n = |(\bigcup \mathcal{V})/\sim_a|$, the following observation (proven in Section 8.8.4.1) gives the rationale behind algorithm 7:

Proposition 8.4.0.1.

$\mathcal{R}_{\preceq_r} = \{K \backslash D \mid \text{for all } i \in \{1, .., n\}, D_i^{\preceq_a} \in \min_\subseteq \text{hs}(\text{pHit}(\mathcal{V}, \preceq_a, i) \backslash (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a})))\}$

In other words, starting with the equivalence class $(\bigcup \mathcal{V})_n^{\preceq}$ of highest ranked axioms of $\bigcup \mathcal{V}$, the axioms discarded from each equivalence class $(\bigcup \mathcal{V})_i^{\preceq}$ must form a minimal hitting set for all elements of $\mathcal{V}$ not hit thus far, and which cannot be hit by any strictly lower ranked axiom.

[22] although possibly less than the canonical minimal hitting set problem mentioned in section 8.1.3.1)

Let hitInt : $2^{\mathcal{L}} \times 2^{2^{\mathcal{L}}} \mapsto 2^{2^{\mathcal{L}}}$ be the function which reduces the sets of an input family $\mathcal{X}$ to their respective intersections with an input set Δ, i.e.:

Definition 8.4.0.4. $\text{hitInt}(\Delta, \mathcal{X}) = \{X \cap \Delta \mid X \in \mathcal{X}\}$

Then algorithm 7 yields $\mathcal{R}_{\preceq_r}$. The family $\mathcal{D}_{\preceq_r}$ is the set of minimal incisions under

Algorithm 7 Prioritized base contraction/revision if $\mathcal{V}$ is known

```
1:  D_{⪯r} ← {∅}
2:  for i ← n to 1 do
3:      D' ← {∅}
4:      PH ← pHit(V, ⪯a, i)
5:      for all D ∈ D_{⪯r} do
6:          V' ← hitInt((⋃V)_i^{⪯a}, PH \ hit(V, D))
7:          for all Γ ∈ min_⊆ hs(V') do
8:              D' ← D' ∪ {D ∪ Γ}
9:          end for
10:     end for
11:     D_{⪯r} ← D'
12: end for
13: R_{⪯r} ← ∅
14: for all D ∈ D_{⪯r} do
15:     R_{⪯r} ← R_{⪯r} ∪ {K \ D}
16: end for
```

construction,[23] and the family $\mathcal{D}'$ is just a temporary variable to avoid concurrent modification of $\mathcal{D}_{\preceq_r}$. The main loop (line 2) iterates over all equivalence classes defined by $\preceq_a$, starting with the best ranked equivalence class, i.e. $(\bigcup \mathcal{V})_n^{\preceq_a}$. Line 4, PH is the set of all elements of $\mathcal{V}$ hit by the current equivalence class $(\bigcup \mathcal{V})_i^{\preceq_a}$, and not by any lower ranked equivalence class. Line 5 starts an iteration over all diagnoses under construction. For each diagnosis under construction D, line 6, $PH \setminus \text{hit}(\mathcal{V}, D)$ is the set of elements of $\mathcal{V}$ hit by the current equivalence class, by no lower ranked

[23] In particular, $\mathcal{D}_{\preceq_r}$ in this section will refer to the state of variable $\mathcal{D}_{\preceq_r}$ at some given moments of the execution.

equivalence class, and not hit by D yet. Then $\mathcal{V}'$ (temporarily) contains these, but reduced to their respective intersections with the current equivalence class. The minimal hitting set procedure $\min_{\subseteq}$ hs is called line 7, and for each returned minimal hitting set Γ for $\mathcal{V}'$, $D \cup \Gamma$ is a possibly new diagnosis under construction. Finally, line 13 to 16, each $R \in \mathcal{R}_{\preceq_r}$ is obtained as the complement in K of some diagnosis $D \in \mathcal{D}_{\preceq_r}$.

In the limit case where all equivalence classes hit distinct elements of $\bigcup \mathcal{V}$, i.e. where all $\mathrm{hit}(\mathcal{V}, (\bigcup \mathcal{V})_i^{\preceq_a})$ are mutually disjoint for $1 \leq i \leq n$ (note that $n = 1$ is a particular subcase), computing $\mathcal{R}_{\preceq_r}$ amounts to computing $\mathcal{R}_{\subseteq}$. But in all other cases, computing $\mathcal{R}_{\preceq_r}$ with algorithm 7 is strictly less expensive than computing $\mathcal{R}_{\subseteq}$. To see this, consider the worst possible scenario, where all elements of $\mathcal{V}$ are mutually disjoint, with $m = |\mathcal{V}|$, and for simplicity, let us assume that they all have the same cardinality k. Then $|\mathcal{R}_{\subseteq}| = k^m$. But if $f_i = \max_{V \in \mathrm{pHit}(\mathcal{V}, \preceq_a, i)} |V \cap (\bigcup \mathcal{V})_i^{\preceq_a}|$ and $q_i = |\,\mathrm{pHit}(\mathcal{V}, \preceq_a, i)|$, then inside each equivalence class $(\bigcup \mathcal{V})_i^{\preceq_a}$, the number of incisions to compute is at most $(f_i)^{q_i}$. Then by hypothesis, for some $1 \leq j < l \leq n$, $\mathrm{hit}(\mathcal{V}, (\bigcup \mathcal{V})_j^{\preceq_a})$ and $\mathrm{hit}(\mathcal{V}, (\bigcup \mathcal{V})_l^{\preceq_a})$ overlap, therefore not only $f_l < k$, but more importantly, $\sum_{i=1}^{n} q_i < m$. Therefore the total number of minimal incisions computed during the execution is inferior to $\prod_{i=1}^{n} (f_i)^{q_i} < \prod_{i=1}^{n} (k)^{q_i} = (k)^{\sum_{i=1}^{n} q_i} < k^m$. An additional (exponential) gain may come line 6 from the fact that reducing the elements of $\mathrm{pHit}(\mathcal{V}, \preceq_a, i) \setminus \mathrm{hit}(\mathcal{V}, D)$ to their respective intersections with $(\bigcup \mathcal{V})_i^{\preceq_a}$ can also reduce their overall number. Finally, for each i from n to 1, the hitting set procedure line 7 is called once per hitting set computed thus far, which may turn out to be costly. Observing that the set $PH \setminus \mathrm{hit}(\mathcal{V}, D)$ may be identical for multiple $D \in \mathcal{D}_{\preceq_r}$, a simple optimization consists in keeping track of the hitting sets computed for each D, which guarantees that the number of calls to the hitting set tree procedure

line 7 is bounded by the smallest value between $|\mathcal{D}_{\preceq_r}|$ and $2^{|\,\mathrm{hitInt}((\bigcup\mathcal{V})_i^{\preceq_a}, PH)|}$.

8.5 Computing $\preceq_a$

In the previous section, it was assumed that a preference relation $\preceq_a$ over the axioms of $\bigcup\mathcal{V}$ (i.e. a ranking of these axioms) was available, and efforts were centered on the computation of $\mathcal{R}_{\preceq_r}$, guided by this relation. This assumption is often made in works dealing with prioritized base revision, where $\preceq_a$ is supposed to be obtained from external confidence scores for axioms, or from a manual review of these axioms. Syntactic criteria have also been proposed to define $\preceq_a$, for instance favoring TBox over ABox axioms (or the opposite), favoring axioms whose signatures contain elements with more syntactic occurrences within $K \cup \Theta$, penalizing axioms based on some syntactic patterns (frequent modeling errors), ... For consistent but incoherent KBs, [Kal06] also used as a ranking criterion the number of consequences of a given syntactic form which would be necessarily lost if an axiom ϕ was discarded. In a sense, this section follows this last intuition, although it does not provide a specific $\preceq_a$ (a possible concrete preference relation is evaluated in section 8.6). Instead, it shows that if $\mathcal{V}$ is known, but not $\mathcal{R}_{\subseteq}$, then for each uncertain axiom $\phi \in \bigcup\mathcal{V}$, two subbases of $K \cup \Theta$ can be computed in time polynomial in $\sum_{V\in\mathcal{V}} |V|$, which respectively reflect what retaining or discarding ϕ necessarily implies.

The intuition is simple, and can be summarized with only two questions. Let $\phi \in \bigcup\mathcal{V}$, i.e. ϕ is involved in the inconsistency of $K \cup \Theta$ but is not part of Θ, and is therefore a candidate for removal. The first question one may ask is, if ϕ was retained, which part of the initial base $K \cup \Theta$ would necessarily be retained with it. Let $\mathcal{S}_\phi$ be the set of all remainders which contain ϕ, i.e.:

Definition 8.5.0.5. $\mathcal{S}_\phi = \{R \in \mathcal{R}_\subseteq \mid \phi \in R\} \cup \Theta$

Then the knowledge necessarily retained together with ϕ is $\bigcap \mathcal{S}_\phi$. Whichever the selected remainders are, if they all contain ϕ, then the output of the process is guaranteed to be at least as strong as $\bigcap \mathcal{S}_\phi$. Or in other words, if one adheres to the assumption made throughout this chapter that all selected subbases should be maximal wrt to set-inclusion, and if additionally one would like to retain ϕ, then $\bigcap \mathcal{S}_\phi$ must be retained as well. In particular, if $\phi_1, \phi_2 \in \bigcup \mathcal{V}$, some properties of $\bigcap \mathcal{S}_{\phi_1}$ and $\bigcap \mathcal{S}_{\phi_2}$ considered as a theory (i.e. $\mathrm{Cn}(\bigcap \mathcal{S}_{\phi_1})$ and $\mathrm{Cn}(\bigcap \mathcal{S}_{\phi_2})$) may be exploited to compare the impact of retaining ϕ_1 to the impact of retaining ϕ_2, setting a basis for a semantically grounded computation of $\preceq_a$ (note that because all elements of $\mathcal{S}_{\phi_i}$ are remainders, they are also consistent, so $\bigcap \mathcal{S}_{\phi_i}$ is consistent as well).

A first legitimate objection to this proposal may be made. Arguably, in order to evaluate the impact of retaining ϕ, considering all elements of $\mathcal{S}_\phi$ individually is more accurate than considering $\bigcap \mathcal{S}_\phi$ only. But if all elements of $\mathcal{S}_\phi$ were known for all ϕ, then from the definition of $\mathcal{S}_\phi$, the whole remainder set $\mathcal{R}_\subseteq$ would be known as well, which is precisely the bottleneck addressed by the proposals made in Section 8.4 and this one. So $\bigcap \mathcal{S}_\phi$ should primarily be viewed as a computational compromise, altogether semantically motivated and easy to obtain if $\mathcal{V}$ is already known, as will be shown below.

A case could also be made for considering the disjunctive KB $\bigvee \mathcal{S}_\phi$ (defined in section 8.1.3.2) instead of $\bigcap \mathcal{S}_\phi$ for this purpose. Arguably, $\bigvee \mathcal{S}_\phi$ is a more accurate representation of the knowledge being retained together with ϕ. But because disjunctive KBs cannot be natively represented in most DLs, $\bigvee \mathcal{S}_\phi$ must be manipulated as a family of KBs, namely $\mathcal{S}_\phi$ (and the corresponding theory is set to be $\bigcap\limits_{\Delta \in \mathcal{S}_\phi} \mathrm{Cn}(\Delta)$, as explained in section 8.1.3.2), which leads back to the previous objection, i.e. the

fact that if $\mathcal{S}_\phi$ was known for each $\phi \in \bigcup \mathcal{V}$, then $\mathcal{R}_\subseteq$ would be known as well.

A second base can also be computed to answer a second dual question, which is what part of the input KB would necessarily remain if ϕ was discarded. This base is $\bigcap \mathcal{S}_{\setminus\phi}$, with $\mathcal{S}_{\setminus\phi}$ defined by :

Definition 8.5.0.6. $\mathcal{S}_{\setminus\phi} = \{R \in \mathcal{R}_\subseteq \mid \phi \notin R\} \cup \Theta$

This gives two KBs $\bigcap \mathcal{S}_\phi$ and $\bigcap \mathcal{S}_{\setminus\phi}$ for each uncertain axiom $\phi \in \bigcup \mathcal{V}$, i.e. $2 \cdot |\bigcup \mathcal{V}|$ KBs in total, which can serve as a basis to evaluate the axioms of $\bigcup \mathcal{V}$, and eventually compute the preference relation $\preceq_a$.

If $\mathcal{R}_\subseteq$ is known, obtaining $\bigcap \mathcal{S}_\phi$ and $\bigcap \mathcal{S}_{\setminus\phi}$ is trivial from their definitions. But a more interesting observation is that even if $\mathcal{V}$ only is known, and not $\mathcal{R}_\subseteq$, then the intersections $\bigcap \mathcal{S}_\phi$ and $\bigcap \mathcal{S}_{\setminus\phi}$ can still be obtained in time polynomial in $\sum_{V \in \mathcal{V}} |V|$, without the need to compute $\mathcal{S}_\phi$ and $\mathcal{S}_{\setminus\phi}$.

To show this, a few additional definitions will be useful. K_s will designate the "safe" part of K, i.e. the axioms of K which are not involved in the inconsistency of $K \cup \Theta$, or equivalently, which do not appear in any element of $\mathcal{V}$.

Definition 8.5.0.7. $K_s = K \setminus \bigcup \mathcal{V}$

Finally, given a family of sets $\mathcal{X}$, and two elements x_1 and x_2, the function hitDiff : $2^{2^\mathcal{L}} \times \mathcal{L} \times \mathcal{L} \mapsto 2^{2^\mathcal{L}}$ returns the elements of $\mathcal{X}$ to which x_1 belongs, but not x_2, i.e.:

Definition 8.5.0.8. $\text{hitDiff}(\mathcal{X}, x_1, x_2) = \text{hit}(\mathcal{X}, \{x_1\}) \setminus \text{hit}(\mathcal{X}, \{x_2\})$

Then the two equalities given by propositions 8.5.0.2 and 8.5.0.4 (proven in sections 8.8.5.1 and 8.8.5.3) give two straightforward procedures to compute $\bigcap \mathcal{S}_\phi$ and $\bigcap \mathcal{S}_{\setminus\phi}$ respectively, for each $\phi \in \bigcup \mathcal{V}$, provided $\mathcal{V}$ is known:

Proposition 8.5.0.2. Let $\phi \in \bigcup \mathcal{V}$.

If $\{\phi\} \in \mathcal{V}$, then $\bigcap \mathcal{S}_\phi = \emptyset$.

Otherwise $\bigcap \mathcal{S}_\phi = K_s \cup \{\phi\} \cup \Theta \cup$

$\{\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\}) \mid \forall V_1 \in \text{hit}(\mathcal{V}, \{\phi'\}), \exists V_2 \in \text{hit}(\mathcal{V}, \{\phi\}) : V_2 \setminus \{\phi\} \subseteq V_1\}$

The first precaution is just a limit case, where no remainder contains ϕ. Otherwise, trivially, $K_s \cup \{\phi\} \cup \Theta \subseteq \bigcap \mathcal{S}_\phi$. The last line indicates that aside from ϕ, no other axiom in $\bigcup \text{hit}(\mathcal{V}, \{\phi\})$, i.e. none of the axioms which appears together with ϕ in some $V \in \mathcal{V}$ is retained in $\in \bigcap \mathcal{S}_\phi$. For the remaining axioms of $\bigcup \mathcal{V}$, i.e. for each $\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\})$, in order to decide whether $\phi' \in \bigcap \mathcal{S}_\phi$, it is sufficient to check for each element V_1 of $\mathcal{V}$ which contains ϕ' if there is another element V_2 of $\mathcal{V}$ which contains ϕ, and such that $V_2 \setminus \{\phi\} \subseteq V_1$.

Let f_1 express the cost of computing $\bigcap \mathcal{S}_\phi$ out of $\mathcal{V}$, as a function of $n = \sum_{V \in \mathcal{V}} |V|$. Then from the following proposition, this operation remains polynomial in n (this is proven in Section 8.8.5.2):

Proposition 8.5.0.3. $f_1(n) = O(n^3)$

The second equality provides an immediate procedure to compute $\bigcap \mathcal{S}_{\setminus\phi}$ if $\mathcal{V}$ is known:

Proposition 8.5.0.4. Let $\phi \in \bigcup \mathcal{V}$.

Then $\bigcap \mathcal{S}_{\setminus\phi} = K_s \cup \Theta \ \cup \{\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\}) \mid$

if $\text{hitDiff}(\mathcal{V}, \phi', \phi) \neq \emptyset$ and $\text{hitDiff}(\mathcal{V}, \phi, \phi') \neq \emptyset$,

then $\forall V_1 \in \text{hitDiff}(\mathcal{V}, \phi', \phi), \forall V_2 \in \text{hitDiff}(\mathcal{V}, \phi, \phi') : ((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta \vdash \bot \}$

Again, trivially, $K_s \cup \ \Theta \subseteq \bigcap \mathcal{S}_{\setminus\phi}$. Then because of the "if/then" condition lines 3 and 4, by default, any $\phi' \neq \phi$ such that $\phi' \in \bigcup \mathcal{V}$ and either $\text{hit}(\mathcal{V}, \{\phi\}) \subseteq \text{hit}(\mathcal{V}, \{\phi'\})$ or $\text{hit}(\mathcal{V}, \{\phi'\}) \subseteq \text{hit}(\mathcal{V}, \{\phi\})$ will be retained as well. Otherwise (i.e. if

$\text{hitDiff}(\mathcal{V}, \phi', \phi) \neq \emptyset$ and $\text{hitDiff}(\mathcal{V}, \phi, \phi') \neq \emptyset$), for ϕ' to be retained, for each element V_1 hit by $\{\phi'\}$ but not by $\{\phi\}$, for each element V_2 hit by $\{\phi\}$ but not by $\{\phi'\}$, $((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta \vdash \bot$ must hold. This very last condition may suggest a consistency check, but it is actually not required. Because $\mathcal{V}$ is known, it is sufficient instead to check whether there is a $V_3 \in \mathcal{V}$ such that $V_3 \subseteq ((V_1 \cup V_2) \setminus \{\phi, \phi'\}))$.

Let f_2 express the cost of computing $\bigcap \mathcal{S}_{\setminus\phi}$ out of $\mathcal{V}$, as a function of $n = \sum_{V \in \mathcal{V}} |V|$. Again, this operation remains polynomial in n (this is proven in Section 8.8.5.4):

Proposition 8.5.0.5. $f_2(n) = O(n^4)$

8.6 Evaluation: prioritized base revision guided by linguistic evidence

8.6.1 Datasets

The datasets used for this evaluation are K_1^{DBP} (8329 axioms) and K_3^{DBP} (776 axioms), both described in Chapter 5 section 5.1.4.

Among the 1437 individuals appearing in K_1^{DBP}, 641 either had no DBpedia label, or were considered as potentially homonymous (based on the heuristics described in Chapter 5 Section 5.3.1). They were not discarded from the KB (which would introduce a bias), but were simply not used as a source of linguistic evidence. For each of the 796 other individuals, 200 web pages were retrieved. Similarly, 25 of the 100 individuals appearing in K_3^{DBP} either had no DBpedia label or were considered as potentially homonymous, and 200 web pages were retrieved for each of the 75 other individuals.

K_1^{DBP} and K_3^{DBP} are initially consistent (and coherent). Two different founda-

tional ontologies were tested to extend them, in order to yield an inconsistent KB, according to the proposal made in Chapter 7. The first foundational ontology was used for K_1^{DBP}, and is described in Chapter 7 Section 7.4.3. It is the categorization tool TMEO used in the Senso Comune project, in order to develop a lexical-ontological resource [JVZ+14]. It will be designated in what follows by TMEO. The second foundational ontology is Proton [TKM05], and was used for K_3^{DBP}. It was selected as an alternative to TMEO, because it is more widespread in the SW community.

For K_1^{DBP} and TMEO, the extension strategy was exactly the one described in Chapter 7. The most frequent (modulo the heuristic described in Chapter 7 Section 7.3.2) entities within K_1^{DBP} (individuals, DL atomic concepts and DL atomic roles) were attached to TMEO only if these attachments were obvious. As a reminder, an attachment is an axiom of the form $A(e)$ for an individual $e \in N_{Ind}(K_1^{DBP})$, $B \sqsubseteq A$ for an atomic concept $B \in N_{Con}(K_3^{DBP})$, and either $\top \sqsubseteq \forall R.A$ or $\exists R.\top \sqsubseteq A$ for an atomic role $R \in N_{Role}(K_1^{DBP})$, where $A \in N_{Con}(\Theta)$ is the most specific concept intuitively verifying the axiom. This yielded a set Δ of 62 attachments, i.e. 62 additional axioms, for a total of 8329 + 62 = 8391 removable axioms. These 8391 axioms will be designated with K_{1+}^{DBP} in what follows. Θ was composed of the 69 axioms of TMEO. For K_{1+}^{DBP} and Θ, there were 98 axioms in $\bigcup \mathcal{V}$, which were not all manually reviewed.

For Proton, the strategy was slightly different, because atomic concepts of DBpedia have already been attached to atomic concepts of Proton by [DKSP10], independently from this work. These attachments were simply reused, without adding any further axiom to K. In particular, individuals and DL atomic roles were not attached by [DKSP10], such that all attachments are of the form $B \sqsubseteq A$, with $B \in N_{Con}(K_3^{DBP})$, and $A \in N_{Con}(\Theta)$. In addition, in order to follow the procedure described in Chapter 7, only the most specific attachments were retained, i.e. if

there were two attachments $B \sqsubseteq A_1$ and $B \sqsubseteq A_2$ among the axioms provided by [DKSP10], and such that $\Theta \vdash A_1 \sqsubseteq A_2$, only $B \sqsubseteq A_1$ was incorporated to K. This yielded a set Δ of 41 additional axioms, for a total of $634 + 41 = 675$ removable axioms. These 675 axioms will be designated with K_{3+}^{DBP} in what follows. The OWL version of Proton used by [DKSP10] does not contain negations though (and neither does K_3^{DBP}). Therefore it was extended for the sake of this evaluation with a small number of disjointness axioms (of the form $A_1 \sqsubseteq \neg A_2$) between most generic concepts (for instance between `Event` and `Location`). A locality-based module ([CGHKS08]) was then extracted from Proton+disjointnesses based on $\text{sig}(\Delta)$, such that Θ was composed of 24 axioms only. For K_{3+}^{DBP} and Θ, there were 76 axioms in $\bigcup \mathcal{V}$. among which 18 were manually identified as erroneous by an ontology expert. An axiom ϕ was considered erroneous iff the meaning of some element of its signature was incompatible with its dominant meaning within $K \cup \Theta$.

For both KBs, for each axiom $\phi \in \mathcal{V}$, $\bigcap S_\phi$ and $\bigcap S_{\backslash\phi}$ were computed as explained in section 8.5, as well as the linguistic compliance scores $\text{comp}(\bigcap S_\phi)$ and $\text{comp}(\bigcap S_{\backslash\phi})$, defined in Chapter 4 Section 4.2.5, and a unique score $h(\phi)$ for ϕ was obtained, defined by $h(\phi) = \text{comp}(\bigcap S_\phi) - \text{comp}(\bigcap S_{\backslash\phi})$. These scores in turn defined the preference relation $\preceq_a$ over $\bigcup \mathcal{V}$, i.e. $\phi_1 \prec_a \phi_2$ iff $h(\phi_1) < h(\phi_2)$, and algorithm 7 was applied to compute $\mathcal{R}_{\preceq_r}$, i.e. to perform prioritized base revision.

8.6.2 Results

Results are given in Table 8.1. The baseline is $\mathcal{R}_\subseteq$, described in section 8.2, i.e. the family of all base remainders which are maximal wrt cardinality.

For readability, statistics are not given for the selected subbases, but for their respective complements in K, i.e. for incisions. So $\mathcal{D}_{\preceq_r}$ designates the family of

respective complements in K of each $R \in \mathcal{R}_{\preceq_r}$, and $\mathcal{D}_{\leq}$ designates the family of respective complements in K of each $R \in \mathcal{R}_{\leq}$.

Column "|.|" gives the number of computed incisions for each configuration, where "." stands for $\mathcal{D}_{\preceq_r}$ or $\mathcal{D}_{\leq}$.

For K_{3+}^{DBP} with Proton, there are only 21 optimal remainders wrt $\preceq_r$, so 21 incisions were computed, i.e. $|\mathcal{D}_{\preceq_r}| = |\mathcal{R}_{\preceq_r}| = 21$. A lower bound of 200 for $|\mathcal{R}_{\subseteq}| = |\mathcal{D}_{\subseteq}|$ was also computed independently. So this number is satisfying from a purely quantitative point of view. And from a purely quantitative point of view still, the baseline yields 64 incisions, i.e. $|\mathcal{D}_{\leq}| = |\mathcal{R}_{\leq}| = 64$.

Similarly, for K_{1+}^{DBP}, a lower bound of 200 for $|\mathcal{R}_{\subseteq}| = |\mathcal{D}_{\subseteq}|$ was computed independently. So again, the number $|\mathcal{D}_{\preceq_r}| = |\mathcal{R}_{\preceq_r}| = 15$ of computed incisions is satisfying. But in this case, the baseline $\mathcal{D}_{\leq}$ yields only 2 incisions, i.e. $|\mathcal{D}_{\leq}| = |\mathcal{R}_{\leq}| = 2$. So in order to make the qualitative comparison between $\mathcal{R}_{\preceq_r}$ and the baseline more meaningful, two additional baselines were computed for K_{1+}^{DBP}, namely $\mathcal{D}_{\leq+1}$ and $\mathcal{D}_{\leq+2}$, which are the families of all incisions of minimal size or minimal size +1 for the former, and of minimal size to minimal size +2 for the latter. This yields $|\mathcal{D}_{\leq+1}| = 14$ and $|\mathcal{D}_{\leq+2}| = 34$, with $\mathcal{D}_{\leq} \subset \mathcal{D}_{\leq+1} \subset \mathcal{D}_{\leq+2}$.

For K_{3+}^{DBP}, computing these additional baselines is useless, because $|\mathcal{D}_{\leq}| > |\mathcal{D}_{\preceq_r}|$ already holds, so there is no $i > 0$ such that $|\mathcal{D}_{\leq+i}|$ is closer to $|\mathcal{D}_{\preceq_r}|$ in terms of number of incisions.

"$\bigcup .$" in Table 8.1 stands for the union of all incisions, i.e. the set of all discarded axioms if the output is set to be $\bigcap \mathcal{R}_{\preceq_r}$ (resp. $\bigcap \mathcal{R}_{\leq}$), whereas "$\bigcap .$" designates the axioms which appear in all incisions, i.e. the axioms which would be necessarily lost if at least one element of $\mathcal{R}_{\preceq_r}$ (resp. $\mathcal{R}_{\leq}$) was selected. Column "Tot" gives the number of axioms in "$\bigcup .$" (resp. "$\bigcap .$"). Then the two "Err" columns give the number or axioms of "$\bigcup .$" (resp. "$\bigcap .$") which were actually erroneous. Precision is

		$\lvert . \rvert$	$\lvert \bigcup . \rvert$				$\lvert \bigcap . \rvert$			
			Tot	Err	P	R	Tot	Err	P	R
K_1^{DBP} with TMEO	$\mathcal{D}_{\preceq_r}$	15	18	9	0.5	NA	0	0	NA	NA
	$\mathcal{D}_{\leq}$	2	5	0	0.0	NA	3	0	0.0	NA
	$\mathcal{D}_{\leq+1}$	14	13	3	0.23	NA	1	0	0.0	NA
	$\mathcal{D}_{\leq+2}$	35	23	6	0.26	NA	0	0	NA	NA
K_3^{DBP} with Proton	$\mathcal{D}_{\preceq_r}$	21	19	12	0.63	0.66	3	3	1.0	0.17
	$\mathcal{D}_{\leq}$	64	17	6	0.35	0.33	5	0	0.0	0.0

Table 8.1: Prioritized base revision: results

given by the two "P" columns, and is the proportion of actually erroneous axioms in $\bigcap .$ or $\bigcup .$ respectively. Recall is given by te two "R" columns, and is the proportion of all axioms identified by the ontology expert in $\bigcup \mathcal{V}$ which are also present in $\bigcap .$ or $\bigcup .$ respectively. Because the whole $\bigcup \mathcal{V}$ was reviewed by the expert for K_{3+}^{DBP} only, recall is given for K_{3+}^{DBP} only.

For K_{3+}^{DBP}, as a reminder, there are 18 actually erroneous axioms among the 76 axioms of $\bigcup \mathcal{V}$. The baseline $\mathcal{D}_{\leq}$ produced 64 candidate incisions, 17 different axioms appear in these 64 incisions, and 6 out of these 17 are actually erroneous. This is not significantly better than a random selection of 17 axioms within $\bigcup \mathcal{V}$ (a Fisher exact test yields a p-value of 1). For $\bigcap \mathcal{D}_{\leq}$, none of the 5 axioms appearing in all 64 incisions are actually erroneous. In this case, the result is worse than what could be expected from a random selection, but again not significantly.

For K_{3+}^{DBP} still, $\mathcal{D}_{\preceq_r}$ contains 21 incisions, but of various sizes (from 11 to 14 axioms). 12 of the 19 axioms of $\bigcup \mathcal{D}_{\preceq_r}$ are erroneous, which is significant better than a random selection of 19 axioms within $\bigcup \mathcal{V}$ (p-value < 0.02). For the intersection, the 3 axioms appearing in all 21 incisions are all actually erroneous (but this is not significant).

So for $K_{3.2}^{DBP}$, the linguistic input combined with prioritized revision had a mean-

ingful (positive) impact, whereas the removal of the smallest possible sets of axioms wrt cardinality (which yielded $\mathcal{D}_{\leq}$) did not.

For K^{DBP}_{1+}, the total number or erroneous axioms in $\bigcup \mathcal{V}$ is unknown, so neither recall nor a comparison to a random selection of axioms in $\bigcup \mathcal{V}$ can be provided. But it can nonetheless be observed that the proportion of actually erroneous axioms in $\bigcup \mathcal{D}_{\preceq_r}$ is higher than in $\bigcup$. for any of the three baselines, and significantly higher (p-value < 0.01) for $\mathcal{D}_{\leq}$ and $\mathcal{D}_{\leq+2}$. On the other hand, no conclusion (either positive or negative) can be drawn from the results for $\bigcap$., because $\bigcap \mathcal{D}_{\preceq_r} = \emptyset$.

8.7 Conclusion

This chapter focused on the identification of (sets of) axioms to be discarded from an input KB $K \cup \Theta$, such that $K \cup \Theta$ is inconsistent, incoherent, or has some already identified undesired consequences, and such that Θ needs to be preserved, i.e. only axioms of K may be discarded in practice. These tasks have been designated with different names in the literature, among which syntax-based of belief base contraction (if $\Theta = \emptyset$) or revision (if $\Theta \neq \emptyset$), but also diagnosis, or simply KB debugging.

The focus throughout this chapter was put on inconsistency for readability, but most observations can be generalized to incoherence and undesired consequences provided minor modifications. Section 8.1 characterized the problem, defining useful notions such as the family $\mathcal{R}_{\subseteq}$ of maximal subbases of K (wrt set inclusion) consistent with Θ, and the family $\mathcal{V}$ of minimal conflicts, i.e. minimal subsets of K inconsistent with Θ. Section 8.1.4 identified two potential issues which are inherent to syntax-based contraction/revision for DLs, namely computational cost, and the number of candidate output subbases of K, ie. the cardinality of the family $\mathcal{R}_{\subseteq}$.

Section 8.2 reviewed some of the proposals made in the literature to address

these two issues. It was argued in particular that discarding sets of axioms which are minimal wrt cardinality rather than (only) wrt set inclusion is a questionable heuristic. The adopted solution for the second issue above, i.e. the number of candidate output subbases of K, is *prioritized base debugging*. Intuitively, given a preference relation $\preceq_a$ over the axioms of K, it consists in prioritizing the removal of least preferred axioms of K wrt $\preceq_a$, until consistency is reached. Different definitions and understandings of prioritized base debugging were reviewed in Section 8.2.2, and the one given by [Neb92] was adopted as the intuitively most satisfying one. The output in this case is a selection $\sigma(\mathcal{R}_\subseteq)$ of elements of $\mathcal{R}_\subseteq$.

Section 8.3 then discussed the relation between the computation of the family $\mathcal{R}_\subseteq$ (or equivalently the family $\mathcal{D}$ of all diagnosis for the inconsistency) on the one hand, and the family $\mathcal{V}$ of minimal conflicts on the other hand, reviewing in details the execution of the two main algorithms used in the literature for these purposes, namely Reiter's algorithm for black-box techniques, and a saturated tableau for glass-box techniques.

Section 8.4 proposed an algorithm which performs prioritized base debugging out of $\mathcal{V}$, i.e. which yields the desired selection $\sigma(\mathcal{R}_\subseteq)$ of elements of $\mathcal{R}_\subseteq$, but without the need to compute the whole $\mathcal{R}_\subseteq$.

Section 8.5 proposed a generic solution in order to obtain the preference relation $\preceq_a$, in the form of two bases S_ϕ and $S_{\setminus\phi}$ which, for each candidate axiom ϕ for removal, represent what part of the input KB would necessarily be retained if ϕ was respectively retained or discarded. In particular, it is was shown that these two bases can be computed in polynomial time, provided $\mathcal{V}$ is known.

Section 8.6 finally evaluated the whole strategy, for the specific case where the compliance (as defined in Chapter 4) of S_ϕ and $S_{\setminus\phi}$ to some automatically gathered linguistic input is used in order to rank candidate axioms for removal, i.e. in order to

produce $\preceq_a$. The strategy was compared to a heuristic based on cardinality (reviewed in Section 8.2) as a baseline, with encouraging results. The input inconsistent KBs for this evaluation were produced by application of the manual extension strategy presented in Chapter 7.

8.8 Proofs

8.8.1 Section 8.1

8.8.1.1 Proposition 8.1.3.1

Proposition. $\mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \bigcap_{R \in \mathcal{R}_{\subseteq}} \mathrm{Cn}(R \cup \Theta)$

Proof. Take any $R \in \mathcal{R}_{\subseteq}$. Then $\bigcap \mathcal{R}_{\subseteq} \subseteq R$. So $\bigcap \mathcal{R}_{\subseteq} \cup \Theta \subseteq R \cup \Theta$, and by monotonicity, $\mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \mathrm{Cn}(R \cup \Theta)$. So for all $R \in \mathcal{R}_{\subseteq}$, we have $\mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \mathrm{Cn}(R \cup \Theta)$, such that $\mathrm{Cn}(\bigcap \mathcal{R}_{\subseteq} \cup \Theta) \subseteq \bigcap_{R \in \mathcal{R}_{\subseteq}} \mathrm{Cn}(R \cup \Theta)$.

□

8.8.1.2 Proposition 8.1.4.1

Proposition. $\mathcal{R}_{\subseteq} = \{R' \cap K \mid R' \in \mathcal{R}'_{\subseteq} \text{ and } \Theta \subseteq R'\}$

Proof. For the left inclusion, Let $R \in \mathcal{R}_{\subseteq}$. We need to show that there is an $R' \in \mathcal{R}'_{\subseteq}$ such that $\Theta \subseteq R'$ and $R = R' \cap K$. Take $R' = R \cup \Theta$. Then $\Theta \subseteq R'$, and because from the definition of $\mathcal{R}_{\subseteq}$, we have $R' \subseteq K \cup \Theta$. From the definition of $\mathcal{R}_{\subseteq}$ still, for any $\phi \in K \setminus R$, $R \cup \Theta \cup \{\phi\} \vdash \bot$, and therefore $R' \cup \{\phi\} \vdash \bot$. So from the definition of $\mathcal{R}'_{\subseteq}$, $R' \in \mathcal{R}'_{\subseteq}$.

For the right inclusion, let $R' \in \mathcal{R}'_{\subseteq}$ such that $\Theta \subseteq R'$, and let $R = R' \cap K$. Because $R' \in \mathcal{R}'_{\subseteq}$, $R' \subseteq R \cup \Theta$, and so $R' = (R' \cap K) \cup (R' \cap \Theta) = R \cup \Theta$. In addition, for all $\phi \in (R \cup \Theta) \setminus R'$, $R' \cup \{\phi\} \vdash \bot$, and therefore $R \cup \Theta \cup \{\phi\} \vdash \bot$. So from the definition of $\mathcal{R}_{\subseteq}$, $R \in \mathcal{R}_{\subseteq}$. □

8.8.2 Section 8.2

8.8.2.1 Proposition 8.2.2.2

Proposition. $\mathcal{R}_{\preceq_r} \subseteq \mathcal{R}_{\subseteq}$

Proof. By contraposition. We will show that if $R \notin \mathcal{R}_{\subseteq}$, then $R \notin \mathcal{R}_{\preceq_r}$. Let $R \notin \mathcal{R}_{\subseteq}$. If $R \cup \Theta \vdash \bot$, then $R \notin \mathcal{R}$, and because $\mathcal{R}_{\preceq_r} = \max_{\preceq_r} \mathcal{R}$, $R \notin \mathcal{R}_{\preceq_r}$. If $R \cup \Theta \nvdash \bot$ instead, because $R \notin \mathcal{R}_{\subseteq}$, from the definition of $\mathcal{R}_{\subseteq}$, there must be an $R' \in \mathcal{R}$ such that $R \subset R'$, and therefore for $1 \leq i \leq n$, $R_i^{\preceq_a} \subseteq R_i'^{\preceq_a}$ must hold. In addition, because $R'/\!\sim_a$ is a partition of R', there must be some i such that $R_i^{\preceq_a} \subset R_i'^{\preceq_a}$. Take the smallest i verifying this. Then it holds that $R_i^{\preceq_a} \subset R_i'^{\preceq_a}$, and that $\forall j_{1 \leq j < i} : R_j^{\preceq_a} = R_j'^{\preceq_a}$, therefore from the definition of $\preceq_r$, $R \prec_r R'$ holds, and so $R \notin \max_{\preceq_r} \mathcal{R} = \mathcal{R}_{\preceq_r}$. □

8.8.2.2 Proposition 8.2.2.1

Proposition. $\mathcal{R}_{\preceq_q} \subseteq \mathcal{R}_{\subseteq}$

Proof. If proposition 8.2.2.2 holds, in order to prove proposition 8.2.2.1, it is sufficient by the transitivity of $\subseteq$ to prove that $\mathcal{R}_{\preceq_q} \subseteq \mathcal{R}_{\preceq_r}$. We will show that if $R \notin \mathcal{R}_{\preceq_r}$, then $R \notin \mathcal{R}_{\preceq_q}$. Let $R \notin \mathcal{R}_{\preceq_r}$. If $R \notin \mathcal{R}$, because $\mathcal{R}_{\preceq_q} = \max_{\preceq_q} \mathcal{R}$, $R \notin \mathcal{R}_{\preceq_q}$. If $R \in \mathcal{R}$ but $R \notin \mathcal{R}_{\preceq_r}$, there exist i and $R' \in \mathcal{R}$ such that $\forall j_{1 \leq j < i} : R_j^{\preceq_a} = R_j'^{\preceq_a}$, and $R_i^{\preceq_a} \subset R_i'^{\preceq_a}$. Then $\forall j_{1 \leq j < i} : |R_j^{\preceq_a}| = |R_j'^{\preceq_a}|$, and $|R_i^{\preceq_a}| < |R_i'^{\preceq_a}|$, so $R \prec_q R'$, therefore $R \notin \mathcal{R}_q$. □

8.8.3 Section 8.3

8.8.3.1 Notation

This section introduces additional notation which will be useful for the proofs of the following sections. It also shows why $h(N)$ is uniquely defined for each node $N \in \mathcal{N}^i$, as an immediate consequence of lemma 8.8.3.1 below.

If $\mathcal{G}^i =< \mathcal{N}^i, \mathcal{E}^i, \text{lab}^i >$ is the state of a graph $\mathcal{G}$ after i iterations over the main loop of algorithm 6 for a given input $< K, \Theta >$, then $\mathcal{G}^{i.j}$ will designate the state of $\mathcal{G}$ after after i iteration over the main loop and j iterations over the inner loop line 8, such that $\mathcal{G}^i$ and $\mathcal{G}^{i.0}$ are two alternative denominations from the same state of $\mathcal{G}$. A similar notation will be used for $\mathcal{N}^{i.j}$, $\mathcal{E}^{i.j}$, etc.

In addition, for each node $N \in \mathcal{N}$, a path P from the root node N_{root} to N will be represented as an indexed list of nodes $P = (N_{root}, .., N)$, with $|P|$ the number of nodes in P, and $P(k)$ will designate the $(k-1)^{th}$ node in this path starting from the root, i.e. $P(0) = N_{root}$, and $P(|P|-1) = N$. If $0 \leq i \leq j \leq |P|-1$, then $P(i,j)$ will designate the subpath $(P(i), .., P(j))$. The extension of a path P with a node N will be designated with $\text{ext}(P, N)$, i.e. if $P = (N_{root}, .., N_1)$, then $\text{ext}(P, N_2) = (N_{root}, .., N_1, N_2)$. Finally, $\text{lab}(P)$ will designate the set of axioms labeling the edges of P, i.e. $\text{lab}(P) = \bigcup_{k=1}^{|P|-1} \text{lab}(P(k-1), P(k))$.

At any step of the execution, $\mathcal{G}$ is a DAG, therefore no node can appear twice in a path P. And because N_{root} is the unique root node of $\mathcal{G}$, for each $N \in \mathcal{N}$, there is a path P from N_{root} to N. But there may be several of them. The set of all paths from N_{root} to a given $N \in \mathcal{N}$ will be designated with $\mathcal{P}_N$, with $\mathcal{P}_{N_{root}} = \{(N_{root})\}$ by convention, and $\mathcal{P}$ will designate all paths to any $N \in \mathcal{N}$, i.e. $\mathcal{P} = \{\mathcal{P}_N \mid N \in \mathcal{N}\}$.

A first immediate observation from algorithm 6 is that the label of an edge is

never modified during the execution (but an edge can be deleted from the graph). In other words, if $P \in \mathcal{P}^{i.j-1} \cap \mathcal{P}^{i.j}$, then $\text{lab}^{i.j-1}(P) = \text{lab}^{i.j}(P)$ must hold. Therefore the index $i.j$ in $\text{lab}^{i.j}(P)$ can be omitted.

The following lemma states that after each iteration over the main loop of algorithm 6, if N is a node of the graph, then all paths from the root node to N have the same set of labels:

Lemma 8.8.3.1. For each $N \in \mathcal{N}^i$, if $P_1, P_2 \in \mathcal{P}^i_N$, then $\text{lab}(P_1) = \text{lab}(P_2)$

Proof. By induction on i. For the base case $i = 0$, $\mathcal{N}^0 = \{N_{root}\}$, and $\mathcal{P}^0_{N_{root}} = \{P_{N_{root}}\} = \{(N_{root})\}$. So if $P_1, P_2 \in \mathcal{P}^0_{N_{root}}$, then $P_1 = P_2 = P_{N_{root}}$, and $\text{lab}(P_1) = \text{lab}(P_2)$ is trivially verified.

For the inductive case, by IH, for each $N \in \mathcal{N}^{i-1}$, for all $P_1, P_2 \in \mathcal{P}^{i-1}_N$, $\text{lab}(P_1) = \text{lab}(P_2) = h^{i-1}(N)$, such that $h^{i-1}(N)$ is determined (and unique) for N when entering the i^{th} iteration of the main loop.

Then by induction on j (i.e. on the number of iterations over the inner loop line 8), we will show that for each $N \in \mathcal{N}^{i-1.j}$, if $P_1, P_2 \in \mathcal{P}^{i-1.j}_N$, then $\text{lab}(P_1) = \text{lab}(P_2)$.

For the (inner) base case $j = 0$, the property holds by IH for $\mathcal{G}^{i-1} = \mathcal{G}^{i-1.0}$.

For the (inner) inductive case, let ϕ be the axiom of $\text{lab}^{i-1.j-1}(N_1)$ selected line 8 of algorithm 6 during the j^{th} iteration over the inner loop. The first possible case is the one where the condition line 9 is verified, i.e. there is a node $N_2 \in \mathcal{N}^{i-1.j-1}$ such that $h^{i-1.j-1}(N_2) = h^{i-1.j-1}(N_1) \cup \{\phi\}$. In this case, a new edge (N_1, N_2) is added to $\mathcal{E}$ and labeled with ϕ, while the rest of the graph remains unchanged. So no path is deleted, i.e. $\mathcal{P}^{i-1.j-1} \subset \mathcal{P}^{i-1.j}$. Let P be any newly created path, i.e. $P \in \mathcal{P}^{i-1.j} \setminus \mathcal{P}^{i-1.j-1}$, and let $N = P(|P| - 1)$ be the last node in P.

Because the only new edge in $\mathcal{E}^{i-1.j}$ is (N_1, N_2), this new edge must appear in P, otherwise P would not be a new path. So there must be a $0 \leq k < |P| - 1$

such that $P(k) = N_1$ and $P(k+1) = N_2$. If $|P| - 1 = k + 1$, i.e. if $N = N_2$, then $\text{lab}(P) = \text{lab}(P(0,k)) \cup \text{lab}(P(k,k+1)) = h^{i-1.j-1}(N_1) \cup \text{lab}(N_1, N_2) = h^{i-1.j-1}(N_1) \cup \{\phi\}$. And from the condition line 9, $h^{i-1.j-1}(N_1) \cup \{\phi\} = h^{i-1.j-1}(N_2)$, so $\text{lab}(P) = h^{i-1.j-1}(N_2)$. Then because no new node has been created, i.e. $\mathcal{N}^{i-1.j} = \mathcal{N}^{i-1.j-1}$, $N_2 = N \in \mathcal{N}^{i-1.j-1}$ must hold. And because $\mathcal{G}^{i-1.j-1}$ is a DAG with a unique root node, there must be a path P' to $N_2 = N$ in $\mathcal{G}^{i-1.j-1}$, and because $P' \in \mathcal{P}_{N_2}^{i-1.j-1}$, by IH, $\text{lab}(P') = h^{i-1.j-1}(N_2)$ must hold. So there is a path $P' \in \mathcal{P}_N^{i-1.j-1}$ such that $\text{lab}(P') = \text{lab}(P)$.

If $|P| - 1 > k + 1$ instead, i.e. if the last node N of P is not N_2, because the only new edge in $\mathcal{E}^{i-1.j}$ is $(N_1, N_2) = (P(k), P(k+1))$, all other edges in P must be in $\mathcal{E}^{i-1.j-1}$. In particular, for all $k + 1 \leq l < |P| - 1$, $(P(l), P(l+1)) \in \mathcal{E}^{i-1.j-1}$ must hold. Then because no new node has been created, i.e. $\mathcal{N}^{i-1.j} = \mathcal{N}^{i-1.j-1}$, $N_2 \in \mathcal{N}^{i-1.j-1}$ must hold. So there must be a path $P' \in \mathcal{P}_N^{i-1.j-1}$ and a $0 \leq m < |P'| - 1$ such that $P'(m) = N_2$, and $P'(m, |P'| - 1) = P(k+1, |P| - 1)$. So $\text{lab}(P'(m, |P'| - 1)) = \text{lab}(P(k+1, |P| - 1))$, and because $P'(m) = P(k+1) = N_2$, $P'(0, m) \in \mathcal{P}_{N_2}^{i-1.j-1}$, and therefore $\text{lab}(P'(0, m)) = h^{i-1.j-1}(N_2)$. Then from the condition line 9, $h^{i-1.j-1}(N_2) = h^{i-1.j-1}(N_1) \cup \{\phi\} = \text{lab}(P(0, k+1))$. So $\text{lab}(P') = \text{lab}(P'(0,m)) \cup \text{lab}(P'(m, |P'|-1)) = \text{lab}(P(0, k+1)) \cup \text{lab}(P(k+1, |P|-1)) = \text{lab}(P)$. Therefore in both cases ($N = N_2$ or $N \neq N_2$), there is already a $P' \in \mathcal{P}_N^{i-1.j-1}$ such that $\text{lab}(P') = \text{lab}(P)$.

By IH, for all $P_1, P_2 \in \mathcal{P}_N^{i-1.j-1}$, $\text{lab}(P_1) = \text{lab}(P_2)$, and because $P' \in \mathcal{P}_N^{i-1.j-1}$ and $\text{lab}(P') = \text{lab}(P)$, for all $P_3 \in \mathcal{P}_N^{i-1.j-1}$, $\text{lab}(P) = \text{lab}(P_3)$ must hold, so for all $P_1, P_2 \in \mathcal{P}_N^{i-1.j-1} \cup \{P\}$, $\text{lab}(P_1) = \text{lab}(P_2)$ must hold as well. Then by induction on the number of new paths, i.e. on the cardinality of $\mathcal{P}^{i-1.j} \setminus \mathcal{P}^{i-1.j-1}$, for each $N \in \mathcal{N}^{i-1.j} = \mathcal{N}^{i-1.j-1}$, if $P_1, P_2 \in \mathcal{P}_N^{i-1.j}$, then $\text{lab}(P_1) = \text{lab}(P_2)$ must hold.

The second possible case is the one where the condition line 9 is not verified, and

neither is the condition line 12. Then N_1 has no ϕ-successor, and the graph remains unchanged, i.e. $\mathcal{G}^{i-1.j} = \mathcal{G}^{i-1.j-1}$, and because the property holds by IH for $\mathcal{G}^{i-1.j-1}$, it also holds for $\mathcal{G}^{i-1.j}$.

The last possible case is the where the condition line 9 is not verified, but the condition line 12 is, and a a new node N_2 is created as a ϕ-successor for N_1. In this case, the only new edge in $\mathcal{E}^{i-1.j} \setminus \mathcal{E}^{i-1.j-1}$ (provided it is not deleted line 24) is (N_1, N_2). And because N_2 is a fresh node, it has no sucessor in the graph. So if a new path P is created, it is a path to N_2. Therefore for all $N \in \mathcal{N}^{i-1.j} \setminus \{N_2\}$, $\mathcal{P}_N^{i-1.j} \subseteq \mathcal{P}_N^{i-1.j-1}$. Take any $P_1, P_2 \in \mathcal{P}_N^{i.j-1}$. Because $\mathcal{P}_N^{i-1.j} \subseteq \mathcal{P}_N^{i-1.j-1}$, $P_1 \in \mathcal{P}_N^{i-1.j-1}$ and $P_2 \in \mathcal{P}_N^{i-1.j-1}$ must hold, and by IH, $\text{lab}(P_1) = \text{lab}(P_2)$. Finally, because N_2 is a fresh node and $\mathcal{E}^{i-1.j} \setminus \mathcal{E}^{i-1.j-1} \subseteq \{(N_1, N_2)\}$, there is at most one path to N_2 in $\mathcal{P}^{i-1.j}$, i.e. $|\mathcal{P}_{N_2}^{i-1.j}| \leq 1$, so if $P_1, P_2 \in \mathcal{P}_{N_2}^{i-1.j}$, then $P_1 = P_2$, and $\text{lab}(P_1) = \text{lab}(P_2)$ trivially holds. □

As noted above, an immediate consequence of lemma 8.8.3.1 is that $h(N)$ is uniquely defined for each $N \in \mathcal{N}^i$.

8.8.3.2 Proposition 8.3.1.1

Proposition. $\mathcal{V} = \{\text{lab}^n(N) \mid N \in \mathcal{N}^n\} \setminus \{\emptyset\}\}$

8.8.3.2.1 Lemmas

Lemma 8.8.3.2. At any execution step $i.j$, for all $T, T' \in \{\text{lab}(N) \mid N \in \mathcal{N}^{i.j}\} \setminus \{\emptyset\}$, $T \not\subset T'$.

Proof. By induction on the cumulated number k of iterations over the inner loop line 8. For the base case $k = 0$, $\mathcal{N}^{0.0} = \{N_{root}\}$, so $|\{\text{lab}(N) \mid N \in \mathcal{N}^{0.0}\}| \setminus \{\emptyset\} \leq 1$, and the lemma trivially holds.

For the inductive case, as a notational shorcut, let $\mathcal{T}^{i.j-1} = \{\text{lab}(N) \mid N \in \mathcal{N}^{i.j-1}\} \setminus \{\emptyset\}$, and $\mathcal{T}^{i.j} = \{\text{lab}(N) \mid N \in \mathcal{N}^{i.j}\} \setminus \{\emptyset\}$. Then we need to show that for all $T_j, T'_j \in \mathcal{T}^{i.j}$, $T_j \not\subset T'_j$ holds, knowing by IH that for all $T_{j-1}, T'_{j-1} \in \mathcal{T}^{i.j-1}$, $T_{j-1} \not\subset T'_{j-1}$ holds.

Let $N_1 \in \mathcal{N}^{i.j-1}$ be the selected node line 7 during the $(i)^{th}$ iteration over the main loop, and let ϕ be the axiom selected during iteration $i.j$ over the inner loop line 8. If the condition line 9 is verified, or if the condition line 12 is not verified, then no new node is created, and labels of nodes remain unchanged, so $\mathcal{T}^{i.j} = \mathcal{T}^{i.j-1}$, so by IH the property holds for $\mathcal{T}^{i.j}$.

Now let us consider the case where a new node N_2 is created (line 13). If the condition line 15 is verified, i.e. if there is a $N_4 \in \mathcal{N}^{i.j-1}$ which verifies $\text{lab}^{i.j-1}(N_4) \neq \emptyset$ and $\text{lab}^{i.j-1}(N_4) \cap (h(N_1) \cup \{\phi\}) = \emptyset$, then $\text{lab}^{i.j-1}(N_4)$ is used to label N_2, such that $\mathcal{T}^{i.j} = \mathcal{T}^{i.j-1}$, and again, by IH, the property holds for $\mathcal{T}^{i.j}$.

The last possible case is the one where for all $N_4 \in \mathcal{N}^{i.j-1}$ such that $\text{lab}^{i.j-1}(N_4) \neq \emptyset$, $\text{lab}^{i.j-1}(N_4) \cap (h(N_1) \cup \{\phi\}) = \emptyset$ holds. Or in other words, because $h(N_1) \cup \{\phi\} = h(N_2)$, for all $T \in \mathcal{T}^{i.j-1}$, $T \cap (h(N_2)) \neq \emptyset$ holds. Then a label $T_2 = \text{lab}^{i.j}(N_2)$ is returned by the call to the function ComputeConflict line 19. In addition, there is only one call to the function ComputeConflict during the execution of the inner loop $i.j$, i.e. T_2 is the only label possibly created during iteration $i.j$. If $T_2 = \emptyset$, then from the definition of $\mathcal{T}^{i.j-1}$, $T_2 \notin \mathcal{T}^{i.j-1}$, so once again $\mathcal{T}^{i.j} = \mathcal{T}^{i.j-1}$, and the property holds by IH.

If $T_2 \neq \emptyset$, from the definition of the function ComputeConflict, T_2 must be such that $h(N_2) \cap T_2 = \emptyset$. Let us assume by contradiction that $T_1 \subset T_2$ for some $T_1 \in \mathcal{T}^{i.j}$. If $T_1 \notin \mathcal{T}^{i.j-1}$, then T_1 must have been created during execution $i.j$, and because T_2 is the only label possibly created during iteration $i.j$, $T_1 = T_2$ must hold, which contradicts $T_1 \subset T_2$. If $T_1 \in \mathcal{T}^{i.j-1}$, because $T_1 \subset T_2$ and $h(N_2) \cap T_2 = \emptyset$,

$T_1 \cap h(N_2) = \emptyset$ must hold, but this contradicts the above observation that for all $T \in \mathcal{T}^{i.j-1}$, $T \cap (h(N_2)) \neq \emptyset$. So if T_2 is the (only) new element in $\mathcal{T}^{i.j} \setminus \mathcal{T}^{i.j-1}$, then there is no $T_1 \in \mathcal{T}^{i.j}$ such that $T_1 \subset T_2$.

We can now show that the other direction holds as well, i.e that if T_2 is the (only) new element in $\mathcal{T}^{i.j} \setminus \mathcal{T}^{i.j-1}$, there is no $T_1 \in \mathcal{T}^{i.j}$ such that $T_2 \subset T_1$. Let us assume by contradiction that $T_2 \subset T_1$ for some $T_1 \in \mathcal{T}^{i.j}$. If $T_1 \notin \mathcal{T}^{i.j-1}$, then T_1 must have been created during execution $i.j$, and because T_2 is the only label possibly created during iteration $i.j$, $T_1 = T_2$ must hold, which contradicts $T_2 \subset T_1$. If $T_1 \in \mathcal{T}^{i.j-1}$, then from the definition of $\mathcal{T}^{i.j-1}$, $T_1 = \text{lab}(N')$ for some $N' \in \mathcal{N}^{i.j-1}$. And because $\text{lab}(N_2) = T_2$ and $T_2 \subset T_1$, the condition line 22 is verified for $N_5 = N'$, and so line 26, $\text{lab}(N') = \text{lab}(N_5)$ is replaced by T_2. So by induction on the number of nodes in $\mathcal{N}^{i.j-1}$ which are labeled with T_1, after termination of the loop $i.j$, we have $T_1 \notin \mathcal{T}^{i.j}$, which contradicts the hypothesis. So if T_2 is the (only) new element in $\mathcal{T}^{i.j} \setminus \mathcal{T}^{i.j-1}$, there is no $T_1 \in \mathcal{T}^{i.j}$ such that $T_2 \subset T_1$.

Now let $T, T' \in \mathcal{T}^{i.j}$. If $T = T_2$ or $T' = T_2$, from the above observations, we have $T \not\subset T'$. If $T \neq T_2$ and $T' \neq T_2$, because $\mathcal{T}^{i.j} \setminus \mathcal{T}^{i.j-1} = \{T_2\}$, $\{T, T'\} \subseteq \mathcal{T}^{i.j-1}$ must hold, and by IH $T \not\subset T'$ must hold too. □

8.8.3.2.2 Main proposition

Left inclusion ($\subseteq$). We need to show that $\mathcal{V} \subseteq \{\text{lab}^n(N) \mid N \in \mathcal{N}^n \setminus \{\emptyset\}\}$. As a notational shortcut, let $\mathcal{T}^n = \{\text{lab}(N)^n \mid N \in \mathcal{N}^n \setminus \{\emptyset\}\}$, such that we need to show that $\mathcal{V} \subseteq \mathcal{T}^n$.

If $K \cup \Theta \nvdash \bot$, then $\mathcal{V} = \emptyset$, so trivially, $\mathcal{V} \subseteq \mathcal{T}^n$.

If $K \cup \Theta \vdash \bot$, let $V \in \mathcal{V}$, and let us assume by contradiction that $V \notin \mathcal{T}^n$.

The function ComputeConflict in algorithm 6 returns either a conflict or $\emptyset$,

such that for each $N \in \mathcal{N}^n$, either $\text{lab}^n(N)$ is a conflict, or $\text{lab}(N) = \emptyset$. In addition, because $V \in \mathcal{V}$, V is minimal wrt $\subseteq$ among all conflicts, so for each $N \in \mathcal{N}^n$, either $\text{lab}(N) = \emptyset$ or $\text{lab}(N) \not\subset V$ must hold. In other words, for each $N \in \mathcal{N}^n$, either $\text{lab}(N) = \emptyset$ or there is a $\phi \in \text{lab}(N)$ such that $\phi \notin V$.

Now select a path in $\mathcal{P}^n$ by applying the following procedure. Let N and P be two variables, initialized with $N \leftarrow N_{root}$ and $P \leftarrow (N_{root})$. At any step, if $\text{lab}^n(N) = \emptyset$, then P is the selected path. Otherwise, select one $\phi \in \text{lab}^n(N) \setminus V$. If N has no ϕ-successor, then P is the selected path. If N has a ϕ-successor N_1, then $N \leftarrow N_1$, $P \leftarrow \text{ext}(P, N_1)$, and the operation is repeated for $N = N_1$. Because each node of the graph can appear only once in P, and because $\mathcal{N}^n$ is finite, the procedure must terminate. After termination, each $\phi \in \text{lab}^n(P)$ is such that $\phi \notin V$, or in other words $h(N) \cap V = \emptyset$. So from the definition of $\mathcal{D}$, $h(N) \notin \mathcal{D}$ must hold. If $\text{lab}^n(N) = \emptyset$, this contradicts theorem 8.3.1.1. If $\text{lab}^n(N) \neq \emptyset$ but N has no successor, the only possible reason is given by line 12 of algorithm 6: there is an execution step i and a node $N_3 \in \mathcal{N}_i$ such that $\text{lab}^i(N_3) = \emptyset$ and $h(N_3) \subset h(N) \cup \{\phi\}$. But $\text{lab}^i(N_3) = \emptyset$ iff $h(N_3)$ is an incision, i.e. iff $(K \setminus h(N_3)) \cup \Theta \nvdash \bot$. And because $h(N_3) \subset h(N)$, $(K \setminus h(N)) \subset (K \setminus h(N_3))$, so by monotonicity $(K \setminus h(N)) \cup \Theta \nvdash \bot$ must hold as well. Therefore $h(N)$ must be an incision too, which implies $h(N) \cap V \neq \emptyset$, and contradicts $h(N) \cap V = \emptyset$ above.

Right inclusion ($\supseteq$). We need to show that $\mathcal{V} \supseteq \{\text{lab}^n(N) \mid N \in \mathcal{N}^n \setminus \{\emptyset\}\}$. Again, as a notational shortcut, let $\mathcal{T}^n = \{\text{lab}^n(N) \mid N \in \mathcal{N}^n \setminus \{\emptyset\}\}$, such that we need to show that $\mathcal{V} \supseteq \mathcal{T}^n$.

If $K \cup \Theta \nvdash \bot$, then $\mathcal{V} = \emptyset$, and $\mathcal{N}^n = \{N_{root}\}$, with $\text{lab}^n(N_{root}) = \{\emptyset\}$, so $\mathcal{T}^n = \{\emptyset\} \setminus \{\emptyset\} = \emptyset$, and $\mathcal{V} \supseteq \mathcal{T}^n$ trivially holds.

If $K \cup \Theta \vdash \bot$, let $T \in \mathcal{T}^n$, and let us assume by contradiction that $T \notin \mathcal{V}$.

From algorithm 6, a label can only be introduced line 19, from a call to the function ComputeConflict. So if $N \in \mathcal{N}^n$, from the definition of function ComputeConflict, either $\text{lab}^n(N) \neq \emptyset$ or $\text{lab}^n(N)$ is a conflict, and because $T \in \mathcal{T}^n$, $T \neq \emptyset$, T must be a conflict for K and Θ. Then because $\mathcal{V}$ is the family of all minimal conflicts for K and Θ, there must be a $V \in \mathcal{V}$ such that $V \subseteq T$, and from lemma 8.8.3.2, $V \not\subset T$, such that $T = V$ must hold, and therefore $T \in \mathcal{V}$, contradicting the hypothesis.

8.8.3.3 Proposition 8.3.1.2

Proposition. Let $\mathcal{G}_1^i =< \mathcal{N}, \mathcal{E}, \text{lab}_1 >$ be a possible state of a graph built during an execution of algorithm 6 for $< K_1, \Theta_1 >$, such that $\mathcal{D}_{<K_1,\Theta_1>} = \{h_{\mathcal{G}_1}(N) \mid N \in \mathcal{N}$ and $\text{lab}_1(N) = \emptyset\}$, but there is a $V \in \mathcal{V}_{<K_1,\Theta_1>} \setminus \{\text{lab}_1(N) \mid N \in \mathcal{N}\}$. Then for any $q_1 \in \text{mapInt}(K_1)$, there is a $< K_2, \Theta_2 >$, a $q_2 \in \text{mapInt}(K_2)$ and a possible state $\mathcal{G}_2^k =< \mathcal{N}, \mathcal{E}, \text{lab}_2 >$ of a graph for $< K_2, \Theta_2 >$ such that $< \mathcal{N}, \mathcal{E}, l_{q_1} \circ \text{lab}_1 >= < \mathcal{N}, \mathcal{E}, l_{q_2} \circ \text{lab}_2 >$, but $\mathcal{D}_{<K_2,\Theta_2>} \setminus \{h_{\mathcal{G}_2}(N) \mid N \in \mathcal{N}$ and $\text{lab}_2(N) = \emptyset\} \neq \emptyset$.

Proof. As a notational shortcut, $\mathcal{V}_{<K_i,\Theta_i>}$ (resp. $\mathcal{D}_{<K_i,\Theta_i>}$ and $h_{\mathcal{G}_i}$) will be designated with $\mathcal{V}_i$ (resp. $\mathcal{D}_i$ and h_i).

As a reminder of Section 8.3.1, if $q \in \text{mapInt}(\Gamma)$ for some finite set Γ of axioms, and if $X \subseteq \Gamma$, then $l_q(X) = \{q(\phi) \mid \phi \in X\}$. As a notational shortcut still, if $\mathcal{X} \subseteq 2^\Gamma$, i.e. if $\mathcal{X}$ is a family of sets of axioms, then $l_q(\mathcal{X})$ will designate the family of sets of integers defined by $l_q(\mathcal{X}) = \{l_q(X) \mid X \in \mathcal{X}\}$.

Let $q_1 \in \text{mapInt}(K_1)$, and let $\mathcal{W}$ be the family of sets of integers defined by $\mathcal{W} = \{l_{q_1}(\text{lab}_1(N)) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}$. From lemma 8.8.3.2, for all $N, N' \in \mathcal{N}$ such that $\text{lab}_1(N) \neq \emptyset$, $\text{lab}_1(N) \not\subset \text{lab}_1(N')$ must hold, so because l_{q_1} is injective, for all $W, W' \in \mathcal{W}$, $W \not\subset W'$ must hold as well, and therefore there is an input $< K_2, \Theta_2 >$

and a bijective mapping $q_2 \in \text{mapInt}(K_2)$ such that $l_{q_2}(\mathcal{V}_2) = \mathcal{W}$.

Take any input $< K_2, \Theta_2 >$ and a bijective mapping $q_2 \in \text{mapInt}(K_2)$ such that $l_{q_2}(\mathcal{V}_2) = \mathcal{W}$. Let q_2^- and $l_{q_2}^-$ be the inverse of q_2 and l_{q_2} respectively, and let $\mathcal{G}_2 =< \mathcal{N}, \mathcal{E}, l_{q_2}^- \circ l_{q_1} \circ \text{lab}_1 >$. In other words, $\mathcal{G}_2$ is identical to $\mathcal{G}_1$, but for each $N \in \mathcal{N}$ (resp. for each $E \in \mathcal{E}$), the label of N (resp. of E) in $\mathcal{G}_2$ is $l_{q_2}^-(l_{q_1}(\text{lab}_1(N)))$ (resp. $l_{q_2}^-(l_{q_1}(\text{lab}_1(E)))$).

$\mathcal{G}_2$ is an admissible state of a graph built during an execution of algorithm 6 because it is isomorphic to $\mathcal{G}_1$, i.e. because l_{q_1} and $l_{q_2}^-$ are both injective. In particular, for all $N, N' \in \mathcal{N}$ such that $\text{lab}_2(N) \neq \emptyset$, $\text{lab}_2(N) \not\subset \text{lab}_2(N')$ must hold, because $\text{lab}_1(N) \not\subset \text{lab}_1(N')$ holds from 8.8.3.2, and because $\text{lab}_2(N) = l_{q_2}^-(l_{q_1}(\text{lab}_1(N)))$ and $\text{lab}_2(N') = l_{q_2}^-(l_{q_1}(\text{lab}_1(N')))$. Similarly, for each $N \in \mathcal{N}$, $h_2(N) \cap \text{lab}_2(N) = \emptyset$ must hold.

So in order to show that $\mathcal{G}_2$ is an admissible state of a graph for $< K_2, \Theta_2 >$ in particular, it is sufficient to show that $\mathcal{G}_2$ is admissible for $\mathcal{V}_2$ (or equivalently, for $\mathcal{D}_2$), and more precisely that for each $N \in \mathcal{N}$, either $\text{lab}_2(N)$ is a conflict for $< K_2, \Theta_2 >$ or $\text{lab}_2(N) = \emptyset$, and $\text{lab}_2(N) = \emptyset$ iff $h_2(N)$ is an incision for $\mathcal{V}_2$.

Take any $N \in \mathcal{N}$. If $\text{lab}_2(N) \neq \emptyset$, then from the definition of $\mathcal{W}$, there is a $W \in \mathcal{W}$ such that $\text{lab}_2(N) = l_{q_2}^-(W)$, and because $\mathcal{W} = l_{q_2}(\mathcal{V}_2)$ and l_{q_2} is injective, $l_{q_2}^-(\mathcal{W}) = \mathcal{V}_2$ holds as well, and therefore $l_{q_2}^-(W) = \text{lab}_2(N) \in \mathcal{V}_2$, so $\text{lab}_2(N)$ is a (minimal) conflict for $< K_2, \Theta_2 >$.

So we only have show that $\text{lab}_2(N) = \emptyset$ iff $h_2(N)$ is an incision for $\mathcal{V}_2$. For the left direction, we need to show that if $\text{lab}_2(N) = \emptyset$, then $h_2(N)$ is an incision for $\mathcal{V}_2$. Take any $N \in \mathcal{N}$ such that $\text{lab}_2(N) = \emptyset$. Because $\text{lab}_2(N) = l_{q_2}^-(l_{q_1}(\text{lab}_1(N))))$, and because both l_{q_1} and $l_{q_2}^-$ are bijections, $\text{lab}_1(N) = \emptyset$ must hold. Then because $\mathcal{G}_1^i$ is a graph for $< K_1, \Theta_1 >$, $h_1(N)$ must be an incision for $\mathcal{V}_1$. So for all $V \in \mathcal{V}_1$, $h_1(N) \cap V \neq \emptyset$. In addition, because $\mathcal{G}_1^i$ is a graph for $< K_1, \Theta_1 >$, for all $N' \in \mathcal{N}$,

if $\text{lab}_1(N') \neq \emptyset$, then $\text{lab}_1(N')$ is a conflict for $< K_1, \Theta_1 >$. And because $\mathcal{V}_1$ is the family of all minimal conflicts for $< K_1, \Theta_1 >$, there must be a $V \in \mathcal{V}_1$ such that $V \subseteq \text{lab}_1(N')$. So for all all $N' \in \mathcal{N}$, if $\text{lab}_1(N') \neq \emptyset$, $h_1(N) \cap \text{lab}_1(N') \neq \emptyset$. Or in other words, $h_1(N)$ is an incision for $\{\text{lab}_1(N) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}$. Then as both l_{q_1} and $l^-_{q_2}$ are bijections, $l^-_{q_2}(l_{q_1}(h_1(N))) = h_2(N)$ is an incision for $\{l^-_{q_2}(l_{q_1}(\text{lab}_1(N))) \mid N \in \mathcal{N}\} \setminus \{\emptyset\} = \mathcal{V}_2$.

For the right direction now, we need to show that for each $N \in \mathcal{N}$, if $h_2(N)$ is an incision for $\mathcal{V}_2$, then $\text{lab}_2(N) = \emptyset$. Or by contraposition, we need to show that for each $N \in \mathcal{N}$, if $\text{lab}_2(N) \neq \emptyset$, then $h_2(N)$ is not an incision for $\mathcal{V}_2$. $l^-_{q_1}$ will designate the inverse of the (bijective) mapping l_{q_1}. Take any $N \in \mathcal{N}$ such that $\text{lab}_2(N) \neq \emptyset$. Then because $l^-_{q_1}$ and l_{q_2} are injective, $l^-_{q_1}(l_{q_2}(\text{lab}_2(N))) = \text{lab}_1(N) \neq \emptyset$. And because $\mathcal{G}^i_1$ is a graph for $< K_1, \Theta_1 >$, from the construction of $\mathcal{G}^i_1$, $\text{lab}_1(N) \subseteq K_1 \setminus h_1(N)$, and therefore $h_1(N) \cap \text{lab}_1(N) = \emptyset$ holds. So because $l^-_{q_2}$ and l_{q_1} are injective, $l^-_{q_2}(l_{q_1}(h_1(N))) \cap l^-_{q_2}(l_{q_1}(\text{lab}_1(N)))$ must hold as well, i.e. $h_2(N) \cap \text{lab}_2(N) = \emptyset$. Now because $\mathcal{V}_2 = l^-_{q_2}(\mathcal{W}) = l^-_{q_2}(\{l_{q_1}(\text{lab}_1(N)) \mid N \in \mathcal{N}\} \setminus \{\emptyset\})$, $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N))) \in \mathcal{V}_2$ must hold, and as $h_2(N) \cap \text{lab}_2(N) = \emptyset$, $h_2(N)$ is not an incision for $\mathcal{V}_2$.

So we have shown that $\mathcal{G}_2$ is a possible state of a graph built during an execution of algorithm 6 for $< K_2, \Theta_2 >$. Now we need to show that there is a $D \in \mathcal{D}_2$ such that $D \notin \{h_2(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_2(N) = \emptyset\}$. By hypothesis, there is a $V \in \mathcal{V}_1 \setminus \{\text{lab}_1(N) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}$. So for each $N \in \mathcal{N}$, $\text{lab}_1(N) \neq V$. In addition, if $\text{lab}_1(N) \neq \emptyset$, $\text{lab}_1(N)$ must be a conflict for $< K_1, \Theta_1 >$, and because $V \in \mathcal{V}_1$, V is a minimal conflict for $< K_1, \Theta_1 >$, so $\text{lab}_1(N) \subset V$ cannot hold. Therefore for each $N \in \mathcal{N}$, either $\text{lab}_1(N) \neq \emptyset$, or there is a $\phi \in \text{lab}_1(N) \setminus V$. So there must be at least one incision Δ for $\{\text{lab}_1(N) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}$ such that $\Delta \cap V = \emptyset$, and because $l^-_{q_2}$ and l_{q_1} are injective, $l^-_{q_2}(\{l_{q_1}(\Delta))$ is also an incision for

$l^-_{q_2}(\{l_{q_1}(\{\text{lab}_1(N) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}) = \mathcal{V}_2$. Therefore there must be a $D \in \mathcal{D}_2$ such that $D \subseteq l^-_{q_2}(l_{q_1}(\Delta))$. We will show that $D \notin \{h_2(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_2(N) = \emptyset\}$. By contradiction, let us assume that $D \in \{h_2(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_2(N) = \emptyset\}$. There there is a $N \in \mathcal{N}$ such that $h_2(N) = D$. And because $\mathcal{G}_2$ is a possible state of a graph for $< K_2, \Theta_2 >$ and $h_2(N) = D$ is a (minimal) incision for $\mathcal{V}_2$, $\text{lab}_2(N) = \emptyset$ must hold, and therefore $l^-_{q_1}(\{l_{q_2}(\text{lab}_2(N))) = \text{lab}_1(N) = \emptyset$ holds as well. And by hypothesis, because $\text{lab}_1(N) = \emptyset$, $h_1(N) \in \mathcal{D}_1$, such that $h_1(N) \cap V \neq \emptyset$ must hold. But $h_2(N) = D \subseteq l^-_{q_2}(l_{q_1}(\Delta))$, so $l^-_{q_1}(l_{q_2}(h_2(N))) = h_1(N) \subseteq \Delta$. And because $\Delta \cap V = \emptyset$, $h_1(N) \cap V = \emptyset$ must hold, which contradicts $h_1(N) \cap V \neq \emptyset$.

□

8.8.3.4 Proposition 8.3.1.3

Proposition 8.8.3.1. Let $\mathcal{G}^i_1 =< \mathcal{N}, \mathcal{E}, \text{lab}_1 >$ be a possible state of a graph built during an execution of algorithm 6 for $< K_1, \Theta_1 >$, such that $\mathcal{V}_{<K_1,\Theta_1>} = \{\text{lab}_1(N) \mid N \in \mathcal{N}\} \setminus \{\emptyset\}$, but there is a $D \in \mathcal{D}_{<K_1,\Theta_1>} \setminus \{h_{\mathcal{G}_1}(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_1(N) = \emptyset\}$. Then for any $q_1 \in \text{mapInt}(K_1)$, there is a $< K_2, \Theta_2 >$, a $q_2 \in \text{mapInt}(K_2)$ and a possible state $\mathcal{G}^k_2 =< \mathcal{N}, \mathcal{E}, \text{lab}_2 >$ of a graph for $< K_2, \Theta_2 >$ such that $< \mathcal{N}, \mathcal{E}, l_{q_1} \circ \text{lab}_1 >=< \mathcal{N}, \mathcal{E}, l_{q_2} \circ \text{lab}_2 >$, but $\mathcal{V}_{<K_2,\Theta_2>} \setminus \{\text{lab}_2(N) \mid N \in \mathcal{N}\} \neq \emptyset$.

Proof. This proof is relatively similar to the proof of proposition 8.3.1.2, but relies a different family $\mathcal{W}$ of sets of integers. For readability though, it will be fully developed.

As a notational shortcut, $\mathcal{V}_{<K_i,\Theta_i>}$ (resp. $\mathcal{D}_{<K_i,\Theta_i>}$ and $h_{\mathcal{G}_i}$) will be designated with $\mathcal{V}_i$ (resp. $\mathcal{D}_i$ and h_i).

As a reminder of Section 8.3.1, if $q \in \text{mapInt}(\Gamma)$ for some finite set Γ of axioms, and if $X \subseteq \Gamma$, then $l_q(X) = \{q(\phi) \mid \phi \in X\}$. As a notational shortcut still, if $\mathcal{X} \subseteq 2^\Gamma$,

i.e. if $\mathcal{X}$ is a family of sets of axioms, then $l_q(\mathcal{X})$ will designate the family of sets of integers defined by $l_q(\mathcal{X}) = \{l_q(X) \mid X \in \mathcal{X}\}$.

By hypothesis, there is a $D \in \mathcal{D}_1 \setminus \{h_1(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_1(N) = \emptyset\}$. Let $q_1 \in \text{mapInt}(K_1)$, let $k = |K_1| + 1$, and let W be the set of integers defined by $W = l_{q_1}((\bigcup \mathcal{V}_1) \setminus D) \cup \{k\}$. For instance, if $|K_1| = 10$, $l_{q_1}(\mathcal{V}_1) = \{\{1,6\},\{6,3\},\{1,4,5\}\}$ and $D = \{4,6\}$, then $W = \{1,3,5\} \cup \{11\} = \{1,3,5,11\}$. Because $D \in \mathcal{D}_1$, for all $V \in \mathcal{V}_1$, $D \cap V \neq \emptyset$, such that $l_{q_1}(D) \cap l_{q_1}(V) \neq \emptyset$ also holds. And because $l_{q_1}(D) \cap W = \emptyset$, for each $V \in \mathcal{V}_1$, $(l_{q_1}(D) \cap l_{q_1}(V)) \cap W = \emptyset$, so there must be an $i \in l_{q_1}(D) \cap l_{q_1}(V) \subseteq l_{q_1}(V)$ such that $i \notin W$, and therefore $l_{q_1}(V) \not\subseteq W$. Now because $k > |K_1|$, for all $\phi \in K_1$, $q_1(\phi) \neq k$, and because $\bigcup \mathcal{V}_1 \subseteq K_1$, for each $V \in \mathcal{V}_1$, $k \notin l_{q_1}(V)$, and therefore $W \not\subseteq l_{q_1}(V)$.

Now let us consider the family of sets of integers $\mathcal{W} = \{l_{q_1}(\mathcal{V}_1) \cup \{W\}\}$. We just showed that for each $V \in \mathcal{V}_1$, $l_{q_1}(V) \not\subseteq W$ and $W \not\subseteq l_{q_1}(V)$ hold, so $l_{q_1}(V) \neq W$ must hold, and therefore $|\mathcal{W}| = |\mathcal{V}_1| + 1$. In addition, from the definition of $\mathcal{V}_1$, for each $V, V' \in \mathcal{V}_1$, $V \not\subset V'$, so because l_{q_1} is injective, $l_{q_1}(V) \not\subset l_{q_1}(V')$. And because for each $V \in \mathcal{V}_1$, $l_{q_1}(V) \not\subseteq W$ and $W \not\subseteq l_{q_1}(V)$ hold, $l_{q_1}(V) \not\subset W$ and $W \not\subset l_{q_1}(V)$ must hold as well. So for each $W_1, W_2 \in \mathcal{W}$, $W_1 \not\subset W_2$ holds, therefore there is an input $< K_2, \Theta_2 >$ and a bijective mapping $q_2 \in \text{mapInt}(K_2)$ such that $l_{q_2}(\mathcal{V}_2) = \mathcal{W}$.

Take any input $< K_2, \Theta_2 >$ and a bijective mapping $q_2 \in \text{mapInt}(K_2)$ such that $l_{q_2}(\mathcal{V}_2) = \mathcal{W}$. Let q_2^- and $l_{q_2}^-$ be the inverse of q_2 and l_{q_2} respectively, and let $\mathcal{G}_2 =< \mathcal{N}, \mathcal{E}, l_{q_2}^- \circ l_{q_1} \circ \text{lab}_1 >$. In other words, $\mathcal{G}_2$ is identical to $\mathcal{G}_1$, but for each $N \in \mathcal{N}$ (resp. for each $E \in \mathcal{E}$), the label of N (resp. of E) in $\mathcal{G}_2$ is $l_{q_2}^-(l_{q_1}(\text{lab}_1(N)))$ (resp. $l_{q_2}^-(l_{q_1}(\text{lab}_1(E)))$).

$\mathcal{G}_2$ is an admissible state of a graph built during an execution of algorithm 6 because it is isomorphic to $\mathcal{G}_1$, i.e. because l_{q_1} and $l_{q_2}^-$ are both injective. In particular, for all $N, N' \in \mathcal{N}$ such that $\text{lab}_2(N) \neq \emptyset$, $\text{lab}_2(N) \not\subset \text{lab}_2(N')$ must hold, because

$\text{lab}_1(N) \not\subset \text{lab}_1(N')$ holds from 8.8.3.2, and because $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N)))$ and $\text{lab}_2(N') = l^-_{q_2}(l_{q_1}(\text{lab}_1(N')))$. Similarly, for each $N \in \mathcal{N}$, $h_2(N) \cap \text{lab}_2(N) = \emptyset$ must hold.

So in order to show that $\mathcal{G}_2$ is an admissible state of a graph for $< K_2, \Theta_2 >$ in particular, it is sufficient to show that $\mathcal{G}_2$ is admissible for $\mathcal{V}_2$ (or equivalently, for $\mathcal{D}_2$), and more precisely that for each $N \in \mathcal{N}$, either $\text{lab}_2(N)$ is a conflict for $< K_2, \Theta_2 >$ or $\text{lab}_2(N) = \emptyset$, and $\text{lab}_2(N) = \emptyset$ iff $h_2(N)$ is an incision for $\mathcal{V}_2$. To this end, we can first observe that $\mathcal{V}_2 = l^-_{q_2}(\mathcal{W}) = l^-_{q_2}(l_{q_1}(\mathcal{V}_1) \cup \{W\})$, and therefore $l^-_{q_2}(l_{q_1}(\mathcal{V}_1)) \subset \mathcal{V}_2$. Now take any node $N \in \mathcal{N}$ such that $\text{lab}_2(N) \neq \emptyset$. Then $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N)))$, and because $l^-_{q_2}$ and l_{q_1} are injective, $\text{lab}_1(N) \neq \emptyset$ must hold. And by hypothesis, if $\text{lab}_1(N) \neq \emptyset$, then $\text{lab}_1(N) \in \mathcal{V}_1$, so $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N))) \in l^-_{q_2}(l_{q_1}(\mathcal{V}_1))$, and because $l^-_{q_2}(l_{q_1}(\mathcal{V}_1)) \subset \mathcal{V}_2$, $\text{lab}_2(N) \in \mathcal{V}_2$, such that $\text{lab}_2(N)$ must be a (minimal) conflict for $< K_2, \Theta_2 >$.

So we only have show that $\text{lab}_2(N) = \emptyset$ iff $h_2(N)$ is an incision for $\mathcal{V}_2$. For the left direction, we need to show that if $\text{lab}_2(N) = \emptyset$, then $h_2(N)$ is an incision for $\mathcal{V}_2$. Let $N \in \mathcal{N}$ such that $\text{lab}_2(N) = \emptyset$. Then because $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N)))$, and because $l^-_{q_2}$ and l_{q_1} are injective, $\text{lab}_1(N) = \emptyset$ must hold. Now because $\mathcal{G}^i_1$ is a graph for $< K_1, \Theta_1 >$, if $\text{lab}_1(N) = \emptyset$, then $h_1(N)$ must be an incision for $\mathcal{V}_1$. So for all $V_1 \in \mathcal{V}_1$, $h_1(N) \cap V_1 \neq \emptyset$. Then again because $l^-_{q_2}$ and l_{q_1} are injective, for all $V_2 \in l^-_{q_2}(l_{q_1}(\mathcal{V}_1))$, $l^-_{q_2}(l_{q_1}(h_1(N))) \cap V_2 \neq \emptyset$ must hold, i.e. $h_2(N) \cap V_2$ must hold. Then as $\mathcal{V}_2 = l^-_{q_2}(\mathcal{W}) = l^-_{q_2}(l_{q_1}(\mathcal{V}_1) \cup \{W\})$, for each $V_2 \in \mathcal{V}_2 \setminus l^-_{q_2}(W)$, $V_2 \cap h_2(N) \neq \emptyset$, and we only need to show that $h_2(N) \cap l^-_{q_2}(W) \neq \emptyset$ holds as well.

To do this, let us consider D again. By hypothesis, $D \notin \{h_1(N) \mid N \in \mathcal{N} \text{ and } \text{lab}_1(N) = \emptyset\}$. So because $\text{lab}_1(N) = \emptyset$, $h_1(N) \neq D$ must hold. By hypothesis still, D is a minimal incision for $\mathcal{V}_1$, so because $h_1(N)$ is an incision, $h_1(N) \not\subset D$ must hold, which together with $h_1(N) \neq D$ yields $h_1(N) \not\subseteq D$. Or in other words,

there is a $\phi \in h_1(N) \setminus D$, and because $l^-_{q_2}$ and l_{q_1} are injective, there is a $\phi' \in l^-_{q_2}(l_{q_1}(h_1(N))) \setminus l^-_{q_2}(l_{q_1}(D))$, i.e. $\phi' \in h_2(N) \setminus l^-_{q_2}(l_{q_1}(D))$. In addition, immediately from the construction of $\mathcal{G}^i_1$, if $\phi \in h_1(N)$, then there is a node N_2 such that $\phi \in \text{lab}_1(N_2)$. So $\text{lab}_1(N_2) \neq \emptyset$, and therefore by hypothesis $\text{lab}_1(N_2) \in \mathcal{V}_1$, which implies $\phi \in \bigcup \mathcal{V}_1$. Then because $\phi \in h_1(N) \setminus D$, $\phi \notin D$, such that $\phi \in (\bigcup \mathcal{V}_1) \setminus D$. So $l_{q_1}(\phi) \in l_{q_1}((\bigcup \mathcal{V}_1) \setminus D)$, and therefore $l_{q_1}(\phi) \in l_{q_1}((\bigcup \mathcal{V}_1) \setminus D) \cup \{k\} = W$, such that $l^-_{q_2}(l_{q_1}(\phi)) \in l^-_{q_2} l_{q_1}((\bigcup \mathcal{V}_1) \setminus D) \cup \{k\} = l^-_{q_2}(W)$. Then because $\phi \in h_1(N)$, $l^-_{q_2}(l_{q_1}(\phi)) \in l^-_{q_2}(l_{q_1}(h_1(N))) = h_2(N)$, such that $h_2(N) \cap l^-_{q_2}(W) \neq \emptyset$.

For the right direction now, we need to show that for each $N \in \mathcal{N}$, if $h_2(N)$ is an incision for $\mathcal{V}_2$, then $\text{lab}_2(N) = \emptyset$. $l^-_{q_1}$ will designate the inverse of the (bijective) mapping l_{q_1}. Let $N \in \mathcal{N}$ such that $h_2(N)$ is an incision for $\mathcal{V}_2$. Then for each $V_2 \in \mathcal{V}_2$, $h_2(N) \cap V_2 \neq \emptyset$. So for each $l^-_{q_1}(l_{q_2}(V_2)) \in l^-_{q_1}(l_{q_2}(\mathcal{V}_2))$, $l^-_{q_1}(l_{q_2}(h_2(N))) \cap l^-_{q_1}(l_{q_2}(V_2)) \neq \emptyset$, i.e. $h_1(N) \cap l^-_{q_1}(l_{q_2}(V_2)) \neq \emptyset$. And because $\mathcal{V}_1 \subseteq l^-_{q_1}(l_{q_2}(\mathcal{V}_2))$, for each $V_1 \in \mathcal{V}_1$, $h_1(N) \cap V_1 \neq \emptyset$ must hold, i.e. $h_1(N)$ is an incision for $\mathcal{V}_1$. Now because $\mathcal{G}^i_1$ is a (state of a) graph for $< K_1, \Theta_1 >$, if $h_1(N)$ is an incision, then $\text{lab}_1(N) = \emptyset$ must hold, and so $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N))) = \emptyset$ must hold as well.

So we have shown that $\mathcal{G}_2$ is a possible state of a graph built during an execution of algorithm 6 for $< K_2, \Theta_2 >$. Now we need to show that there is a $V_2 \in \mathcal{V}_2$ such that $V \notin \{l_{q_2}(\text{lab}(N)) \mid N \in \mathcal{N}_2\} \setminus \{\emptyset\}$, or equivalently that $l^-_{q_2}(V_2) \notin \{\text{lab}(N) \mid N \in \mathcal{N}_2\} \setminus \{\emptyset\}$. We will show that this holds for $V_2 = l^-_{q_2}(W)$. Because $\mathcal{V}_2 = l^-_{q_2}(\mathcal{W}) = l^-_{q_2}(l_{q_1}(\mathcal{V}_1)) \cup \{W\}) = l^-_{q_2}(l_{q_1}(\mathcal{V}_1) \cup \{l^-_{q_2}(W)\}$, $l^-_{q_2}(W) \in \mathcal{V}_2$ holds. So we only need to show that for all $N \in \mathcal{N}$ such that $\text{lab}_2(N) \neq \emptyset$, $\text{lab}_2(N) \neq l^-_{q_2}(W)$.

Take any $N \in \mathcal{N}$ such that $\text{lab}_2(N) \neq \emptyset$. Then $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(\text{lab}_1(N)))$, and by hypothesis, if $\text{lab}_1(N) \neq \emptyset$, there is a $V_1 \in \mathcal{V}_1$ such that $\text{lab}_1(N) = V_1$, and so $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(V_1))$. In addition, we have shown above (immediately after the definition of $\mathcal{W}$) that for all $V_1 \in \mathcal{V}_1$, $W \neq l_{q_1}(V_1)$. So because $l^-_{q_2}$ and l_{q_1} are

bijections, for all $V_1 \in \mathcal{V}_1$, $l^-_{q_2}(W) \neq l^-_{q_2}(l_{q_1}(V_1))$ must hold, and because $\text{lab}_2(N) = l^-_{q_2}(l_{q_1}(V_1))$ for some $V_1 \in \mathcal{V}_1$, $\text{lab}_2(N) \neq l^-_{q_2}(W)$ must hold. □

8.8.4 Section 8.4

8.8.4.1 Proposition 8.4.0.1

$\mathcal{R}_{\preceq_r} = \{K \backslash D \mid \text{for all } i \in \{1,..,n\}, D_i^{\preceq_a} \in \min_{\subseteq} \text{hs}(\text{pHit}(\mathcal{V}, \preceq_a, i) \backslash (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a})))\}$

8.8.4.2 Left inclusion ($\subseteq$).

By contraposition. We will show that if R is not an element of the set on the right hand side of proposition 8.4.0.1, then $R \notin \mathcal{R}_{\preceq_r}$.

If $R \not\subseteq K$, then $R \notin \mathcal{R}$, and from definition 8.2.2.3, $R \notin \mathcal{R}_{\preceq_r}$.

If $R \subseteq K$, let $D = K \setminus R$. If $D \not\subseteq \bigcup \mathcal{V}$, then from theorem 8.1.3.1, $D \notin \mathcal{D}$, and from lemma 8.1.3.1, $R \notin \mathcal{R}_{\subseteq}$, such that from proposition 8.2.2.2, $R \notin \mathcal{R}_{\preceq_r}$.

If $R \subseteq K$ and $D \subseteq \bigcup \mathcal{V}$, for any $i \in \{1,..,n\}$, let $\mathcal{H}_i = \text{hs}(\text{pHit}(\mathcal{V}, \preceq_a, i) \setminus (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a}))$. If R is not an element of the set on the right hand side of proposition 8.4.0.1 but $R \subseteq K$, then there must be an $i \in \{1,..,n\}$ such that $D_i^{\preceq_a} \notin \min_{\subseteq} \mathcal{H}_i$. Take the smallest i verifying this. If $D_i^{\preceq_a} \notin \min_{\subseteq} \mathcal{H}_i$, then either $D_i^{\preceq_a} \notin \mathcal{H}_i$, or $D_i^{\preceq_a} \in \mathcal{H}_i$ but $D_i^{\preceq_a} \notin \min_{\subseteq} \mathcal{H}_i$.

Let us consider the first case, i.e. $D_i^{\preceq_a} \notin \mathcal{H}_i$. Then there is a $V \in \text{pHit}(\mathcal{V}, \preceq_a, i) \setminus (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a}))$ such that $D_i^{\preceq_a} \cap V = \emptyset$, therefore the three following hold:

- $V \in \text{pHit}(\mathcal{V}, \preceq_a, i)$, so from definition 8.4.0.3
  $V \notin \text{hit}(\mathcal{V}, \bigcup_{1\leq k<i} (\bigcup \mathcal{V})_k^{\preceq_a})$, and because $D \subseteq \bigcup \mathcal{V}$, $\bigcup_{1\leq k<i} D_k^{\preceq_a} \subseteq \bigcup_{1\leq k<i} (\bigcup \mathcal{V})_k^{\preceq_a}$, so
  $V \notin \text{hit}(\mathcal{V}, \bigcup_{1\leq k<i} D_k^{\preceq_a})$

- $V \notin \text{hit}(\mathcal{V}, D_i^{\preceq_a})$
- $V \notin \text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a})$

Then because $\preceq_a$ is a total preorder over K, $|K/\sim_a| = n$ and $D \subseteq K$, we have $(\bigcup_{1\leq k<i} D_k^{\preceq_a})) \cup D_i^{\preceq_a} \cup (\bigcup_{i<j\leq n} D_j^{\preceq_a})) = D$. So from the three previous observations, $V \notin \text{hit}(\mathcal{V}, D)$, and because $V \in \mathcal{V}$, $V \cap D = \emptyset$ must hold. But from the definition of $\mathcal{V}$, $V \subseteq K$ also holds, such that if $V \cap D = \emptyset$, $V \subseteq K \setminus D$ must hold, i.e. $V \subseteq R$. Finally, because $V \in \mathcal{V}$ still, $V \cup \Theta \vdash \bot$ so by monotonicity, $R \cup \Theta \vdash \bot$, so from definition 8.1.3.2, $R \notin \mathcal{R}$, and from definition 8.2.2.3, $R \notin \mathcal{R}_{\preceq_r}$.

Now let us consider the second case, i.e. $D_i^{\preceq_a} \in \mathcal{H}_i$ but $D_i^{\preceq_a} \notin \min_{\subseteq} \mathcal{H}_i$. Then there must be a $\Phi \subset D_i^{\preceq_a}$ such that $\Phi \in \mathcal{H}_i$. Take the set $\Gamma \subseteq K$ defined by $\Gamma_k^{\preceq_a} = D_k^{\preceq_a}$ for $1 \leq k < i$, $\Gamma_i^{\preceq_a} = \Phi$, and $\Gamma_j^{\preceq_a} = (\bigcup \mathcal{V})_j^{\preceq_a}$ for $i < j \leq n$. Then take any $V \in \mathcal{V}$. Because $V \subseteq K$ and $\preceq_a$ is a total preorder over K with $|K/\sim_a| = n$, there must be a $l \in \{1, .., n\}$ such that $V_l^{\preceq_a} \neq \emptyset$. Let $m \in \{1, .., n\}$ be the smallest integer such that $V_m^{\preceq_a} \neq \emptyset$.

As $V \in \mathcal{V}$, $V \subseteq (\bigcup \mathcal{V})$, and therefore $V_m^{\preceq_a} \subseteq (\bigcup \mathcal{V})_m^{\preceq_a}$ holds, so if $m > i$, because $\Gamma_m^{\preceq_a} = (\bigcup \mathcal{V})_m^{\preceq_a}$ and $V_m^{\preceq_a} \neq \emptyset$, $V \cap \Gamma_m^{\preceq_a} \neq \emptyset$ must hold, and so $V \cap \Gamma \neq \emptyset$. If $m \leq i$, then either $V \cap \Gamma_j^{\preceq_a} \neq \emptyset$ for some $j > m$, and $V \cap \Gamma \neq \emptyset$ trivially holds, or $V \cap \Gamma_j^{\preceq_a} = \emptyset$ for all $j > m$, i.e. $V \notin \text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} \Gamma_j^{\preceq_a})$. In this latter case, because m is the smallest integer such that $V_m^{\preceq_a} \neq \emptyset$, $V \in \text{pHit}(\mathcal{V}, \preceq_a, m)$. So $V \in (\text{pHit}(\mathcal{V}, \preceq_a, m) \setminus \text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} \Gamma_j^{\preceq_a})$. We also know that $\Gamma_i^{\preceq_a} = \Phi \in \mathcal{H}_i$, and because i is the smallest integer such that $D_i^{\preceq_a} \notin \min_{\subseteq} \mathcal{H}_i$, for all $k \in \{1, .., i-1\}$, $D_k^{\preceq_a} = \Gamma_k^{\preceq_a} \in \mathcal{H}_k$. So for any possible value of m in $\{1, .., i\}$, $\Gamma_m^{\preceq_a} \in \mathcal{H}_m$ holds, i.e. $\Gamma_m^{\preceq_a} \in \text{hs}(\text{pHit}(\mathcal{V}, \preceq_a, m) \setminus (\text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} \Gamma_j^{\preceq_a})))$. Then because $V \in (\text{pHit}(\mathcal{V}, \preceq_a , m) \setminus \text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} \Gamma_j^{\preceq_a}))$, from the definition of hs, $V \cap \Gamma_m^{\preceq_a}$ must hold, and therefore $V \cap \Gamma \neq \emptyset$. Therefore whichever the value of m is, $V \cap \Gamma \neq \emptyset$ holds.

So for any $V \in \mathcal{V}$, $\Gamma \cap V \neq \emptyset$, such that Γ is an incision, i.e. $(K \setminus \Gamma) \cup \Theta \nvdash \bot$. So if $R' = K \setminus \Gamma$, $R' \in \mathcal{R}$. But because $\Gamma_k^{\preceq_a} = D_k^{\preceq_a}$ for $1 \leq k < i$, we have $R_k'^{\preceq_a} = R_k^{\preceq_a}$, and because $\Gamma_i^{\preceq_a} = \Phi \subset D_i^{\preceq_a}$, we have $R_i'^{\preceq_a} \subset R_i^{\preceq_a}$. Therefore from the definition of $\preceq_r$, $R \prec_r R'$ must hold, which implies that $R \notin \max_{\preceq_r} \mathcal{R}$, and from definition 8.2.2.3, $R \notin \mathcal{R}_{\preceq_r}$.

8.8.4.3 Right inclusion ($\supseteq$).

Let R be an element of the set on the right hand side of proposition 8.4.0.1, and let $D = K \setminus R$. We will first show that $R \in \mathcal{R}$.

Take any $V \in \mathcal{V}$. Because $V \subseteq K$ and $\preceq_a$ is a total preorder over K with $|K/{\sim_a}| = n$, there must be an $i \in \{1, .., n\}$ such that $V_i^{\preceq_a} \neq \emptyset$. Let $m \in \{1, .., n\}$ be the smallest integer such that $V_m^{\preceq_a} \neq \emptyset$. Then $V \in \text{pHit}(\mathcal{V}, \preceq_a, m)$. If $V \in \text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} D_j^{\preceq_a})$, then $V \cap D_j^{\preceq_a} \neq \emptyset$ for some $j > m$, and therefore $V \cap D \neq \emptyset$. If $V \notin \text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} D_j^{\preceq_a})$ instead, then $V \in \text{pHit}(\mathcal{V}, \preceq_a, m) \setminus (\text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} D_j^{\preceq_a}))$, and because $D_m^{\preceq_a} \in \text{hs}(V \in \text{pHit}(\mathcal{V}, \preceq_a, m) \setminus (\text{hit}(\mathcal{V}, \bigcup_{m<j\leq n} D_j^{\preceq_a})))$, from the definition of hs, $V \cap D_m^{\preceq_a} \neq \emptyset$ must hold, and therefore $V \cap D \neq \emptyset$ holds as well. So for any $V \in \mathcal{V}$, $V \cap D \neq \emptyset$ holds, and therefore D is an incision for K and Θ. Then because $R = K \setminus D$, we have $R \in \mathcal{R}$.

Now in order to show that $R \in \mathcal{R}_{\preceq_r}$, from definition 8.2.2.3, we need to show that $R \in \max_{\preceq_r} \mathcal{R}$. Let us assume by contradiction that $R \notin \max_{\preceq_r} \mathcal{R}$. Because $R \in \mathcal{R}$, there must be an $R' \in \mathcal{R}$ such that $R \prec_r R'$. So from the definition of $\preceq_r$, there must be some $i \in \{1, .., n\}$ such that $R_i^{\preceq_a} \subset R_i'^{\preceq_a}$, and for all $j \in \{i+1, .., n\}$, $R_j^{\preceq_a} = R_j'^{\preceq_a}$. Let $D' = K \setminus R'$. Then we must have $(D')_i^{\preceq_a} \subset D_i^{\preceq_a}$, and for all $j \in \{i+1, .., n\}$, $(D')_j^{\preceq_a} = D_j^{\preceq_a}$. So $\text{pHit}(\mathcal{V}, \preceq_a, i) \setminus (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a})) = \text{pHit}(\mathcal{V}, \preceq_a, i) \setminus (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} (D')_j^{\preceq_a}))$. Now let $\mathcal{X} = \text{pHit}(\mathcal{V}, \preceq_a, i) \setminus (\text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} D_j^{\preceq_a}))$. Then

$\mathcal{X} \subseteq \mathcal{V}$, and because R is an element of the right hand side of proposition 8.4.0.1, $D_i^{\preceq_a} \in \min_{\subseteq} \text{hs}(\mathcal{X})$. Therefore for any $\Gamma \subset D_i^{\preceq_a}$, $\Gamma \notin \text{hs}(\mathcal{X})$. And in particular, $(D')_i^{\preceq_a} \notin \text{hs}(\mathcal{X})$. So there must be a $V \in \mathcal{X}$ such that $V \cap (D')_i^{\preceq_a} = \emptyset$. And because $V \in \mathcal{X}$, $V \notin \text{hit}(\mathcal{V}, \bigcup_{i<j\leq n} (D')_j^{\preceq_a}))$, i.e. for all $j \in \{i+1, .., n\}$, $V \cap (D')_j^{\preceq_a} = \emptyset$. Finally, because $V \in \mathcal{X}$ still, $V \in \text{pHit}(\mathcal{V}, \preceq_a, i)$, so for all $k \in \{1, .., i-1\}$, $V \cap (\bigcup \mathcal{V})_k^{\preceq_a} = \emptyset$, and because $V \subseteq \bigcup \mathcal{V}$, $V_k^{\preceq_a} = \emptyset$. Therefore for all $k \in \{1, .., i-1\}$, $V \cap (D')_k^{\preceq_a} = \emptyset$ also holds. Then because $\preceq_a$ is a total preorder over K, $|K/\sim_a| = n$ and $D' \subseteq K$, we have $(\bigcup_{1\leq k<i} (D')_k^{\preceq_a})) \cup (D')_i^{\preceq_a} \cup (\bigcup_{i<j\leq n} (D')_j^{\preceq_a})) = D'$. So from the three considerations above, $V \cap D' = \emptyset$ must hold. Therefore D' is not an incision, and because $R' = K \setminus D'$, $R' \notin \mathcal{R}$, which contradicts $R' \in \mathcal{R}$. □

8.8.5 Section 8.5

8.8.5.1 Proposition 8.5.0.2

Proposition. Let $\phi \in \bigcup \mathcal{V}$.

If $\{\phi\} \in \mathcal{V}$, then $\bigcap S_\phi = \emptyset$.

Otherwise, $\bigcap S_\phi = K_s \cup \{\phi\} \cup \Theta \cup$

$\{\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\}) \mid \forall V_1 \in \text{hit}(\mathcal{V}, \{\phi'\}), \exists V_2 \in \text{hit}(\mathcal{V}, \{\phi\}) : V_2 \setminus \{\phi\} \subseteq V_1\}$

8.8.5.1.1 Lemmas

Lemma 8.8.5.1. If there are $A, B \in 2^K$ such that $A \cup \Theta \nvdash \bot$ and $\forall \gamma \in B : A \cup \{\gamma\} \cup \Theta \vdash \bot$, then there is an $R \in \mathcal{R}_\subseteq$ such that $A \subseteq R$ and $B \cap R = \emptyset$

Proof. R can be built out of A by applying the following simple procedure: initialize R with A, and then for each $\gamma \in K \setminus A$, if $R \cup \{\gamma\} \cup \Theta \nvdash \bot$, extend R with γ. The procedure ends because K is finite, and when it ends, for each $\gamma \in K \setminus R$, there is

an $R' \subseteq R$ such that $R' \cup \{\gamma\} \cup \Theta \vdash \bot$. So by monotonicity $R \cup \{\gamma\} \cup \Theta \vdash \bot$ as well. Therefore there is no $R'' \in 2^K$ such that $R \subset R''$ and $R'' \cup \Theta \nvdash \bot$, and so $R \in \mathcal{R}_\subseteq$.

To show that $B \cap R = \emptyset$, note that $A \subseteq R$, and by hypothesis, and for each $\gamma \in B$, $A \cup \{\gamma\} \cup \Theta \vdash \bot$. So by monotonicity still, $R \cup \{\gamma\} \cup \Theta \vdash \bot$. But because $R \in \mathcal{R}_\subseteq$, from the definition of $\mathcal{R}_\subseteq$, $R \cup \Theta \nvdash \bot$ must hold, so $\gamma \notin R \cup \Theta$, and therefore $\gamma \notin R$. □

Lemma 8.8.5.2. For all $C \subseteq K$, for all $\gamma \in K$, if $C \cup \Theta \nvdash \bot$ and $C \cup \{\gamma\} \cup \Theta \vdash \bot$, then there is a $V \in \text{hit}(\mathcal{V}, \{\gamma\})$ such that $V \setminus \{\gamma\} \subseteq C$.

Proof. If $C \cup \{\gamma\} \cup \Theta \vdash \bot$, then there must be some subset V of $C \cup \{\gamma\}$ such that $V \cup \Theta \vdash \bot$, and such that $\forall V' \subset V : V' \cup \Theta \nvdash \bot$, so $V \in \mathcal{V}$. Now let us assume that $\{\gamma\} \notin V$. Then $V \cup \Theta \subseteq C \cup \Theta$, and because $C \cup \Theta \nvdash \bot$, by monotonicity, $V \cup \Theta \nvdash \bot$, a contradiction. Therefore $\{\gamma\} \in V$, and because $V \in \mathcal{V}$, $V \in \text{hit}(\mathcal{V}, \gamma)$. □

8.8.5.1.2 Main proposition

Proposition. Let $\phi \in \bigcup \mathcal{V}$.

If $\{\phi\} \in \mathcal{V}$, then $\bigcap S_\phi = \emptyset$.

Otherwise, $\bigcap S_\phi = K_s \cup \{\phi\} \cup \Theta \cup$

$\{\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\}) \mid \forall V_1 \in \text{hit}(\mathcal{V}, \{\phi'\}), \exists V_2 \in \text{hit}(\mathcal{V}, \{\phi\}) : V_2 \setminus \{\phi\} \subseteq V_1\}$

Limit case

If $\{\phi\} \in \mathcal{V}$, then $\{\phi\} \cup \Theta \vdash \bot$, and by monotonicity, for any Δ such that $\phi \in \Delta$, $\Delta \cup \Theta \vdash \bot$. Therefore from the definition of $\mathcal{R}_\subseteq$, for all $R \in \mathcal{R}_\subseteq$, $\phi \notin R$, and from the definition of S_ϕ, $S_\phi = \emptyset$, such that $\bigcap S_\phi = \emptyset$.

Main equality, left inclusion ($\subseteq$)

By contraposition. Assume that ϕ' is not an element of the set on the right-hand

side of the equality. Then we need to show that if $S_\phi \neq \emptyset$, there is a $\Delta \in S_\phi$ such that $\phi' \notin \Delta$, or equivalently that there is an $R \in \mathcal{R}_\subseteq$ such that $\phi \in R$ and $\phi' \notin R$.

By hypothesis, $\phi \in \bigcup \mathcal{V}$, $\phi' \notin K_s$, $\phi' \notin \Theta$ and $\phi' \neq \phi$.

The first possible case is the one one where $\phi' \notin \bigcup \mathcal{V} \setminus \bigcup \mathrm{hit}(\mathcal{V}, \{\phi\})$. The first possible subcase is $\phi' \notin \bigcup \mathcal{V}$. But by hypothesis, $\phi' \notin K_s \cup \Theta$. So if $\phi' \notin \bigcup \mathcal{V}$ either, we have $\phi' \notin K_s \cup \bigcup \mathcal{V} \cup \Theta = K \cup \Theta$, i.e. ϕ' is not in the input base. But for any $R \in \mathcal{R}_\subseteq$, $R \subseteq K \cup \Theta$, so $\phi' \notin R$ trivially holds. The second possible subcase is $\phi' \in \bigcup \mathcal{V} \cap \bigcup \mathrm{hit}(\mathcal{V}, \{\phi\})$, which is equivalent to $\phi' \in \bigcup \mathrm{hit}(\mathcal{V}, \{\phi\})$, as $\bigcup \mathrm{hit}(\mathcal{V}, \{\phi\}) \subseteq \bigcup \mathcal{V}$. Because $\phi' \in \bigcup \mathrm{hit}(\mathcal{V}, \{\phi\})$, there is a $V \in \mathrm{hit}(\mathcal{V}, \{\phi\})$ such that $\phi' \in V$. Now set $A = V \setminus \{\phi'\}$, and $B = \{\phi'\}$ Then because $V \in \mathcal{V}$, it holds that $A \cup \Theta \nvdash \bot$ and $A \cup \{\phi'\} \cup \Theta \vdash \bot$. So from lemma 8.8.5.1, there is an $R \in \mathcal{R}_\subseteq$ such that $A \subseteq R$ and $\phi' \notin R$. Then because $V \in \mathrm{hit}(\mathcal{V}, \{\phi\})$ and $\phi' \neq \phi$, $\phi \in V \setminus \{\phi'\} = A \subseteq R$.

The second possible case is the one where $\phi' \in \bigcup \mathcal{V} \setminus \{\phi\}$, and there is a $V_1 \in \mathrm{hitDiff}(\mathcal{V}, \phi', \phi)$ and a $V_2 \in \mathrm{hitDiff}(\mathcal{V}, \phi, \phi')$ such that if $F \doteq V_1 \cup V_2 \setminus \{\phi, \phi'\}$, then $F \cup \Theta \nvdash \bot$. In this case, $V_1 \subseteq F \cup \{\phi'\}$, and because $V_1 \in \mathcal{V}$, $V_1 \cup \Theta \vdash \bot$, so by monotonicity, $F \cup \{\phi'\} \cup \Theta \vdash \bot$. Similarly, because $V_2 \subseteq F \cup \{\phi\}$, $F \cup \{\phi\} \cup \Theta \vdash \bot$ as well. So from lemma 8.8.5.1, replacing A by F and B by $\{\phi, \phi'\}$, there is an $R \in \mathcal{R}_\subseteq$ such that $\phi \notin R$ and $\phi' \notin R$.

Main equality, right inclusion ($\supseteq$)

Let us assume that ϕ' is an element of the set on the right-hand side of the equality, We need to show that $\phi' \in \bigcap S_\phi$, or equivalently, if there is an $R \in \mathcal{R}_\subseteq$ such that $\phi \in R$, we need to show that $\phi' \in R$.

If $\phi' \in K_s \cup \Theta$, then $\phi' \notin \bigcup \mathcal{V}$ from the definitions of K_s and $\mathcal{V}$, so $\mathrm{hit}(\mathcal{V}, \{\phi'\}) = \emptyset$, and from the contraposition of lemma 8.8.5.2, there is no $C \subseteq K$ such that both $C \cup \Theta \nvdash \bot$ and $C \cup \{\phi'\} \cup \Theta \vdash \bot$ hold. In particular, because $R \subseteq K$ and $R \cup \Theta \nvdash \bot$,

$R \cup \{\phi'\} \cup \Theta \nvdash \bot$ cannot hold. But from the definition of $\mathcal{R}_\subseteq$, if $\phi' \notin R$, then $R \cup \{\phi'\} \cup \Theta \vdash \bot$ must hold. So $\phi' \in R$.

If $\phi' = \phi$, because $\phi \in R$, $\phi' \in R$ holds trivially.

So one can focus on the case where $\phi' \notin K_s \cup \Theta \cup \{\phi\}$, but $\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\})$, and $\forall V_1 \in \text{hit}(\mathcal{V}, \{\phi'\}), \exists V_2 \in \text{hit}(\mathcal{V}, \{\phi\}) : V_2 \setminus \{\phi\} \subseteq V_1$.

Take any $V_1 \in \text{hit}(\mathcal{V}, \{\phi'\})$. By hypothesis, there is a $V_2 \in \text{hit}(\mathcal{V}, \{\phi\})$ such that $V_2 \setminus \{\phi\} \subseteq V_1$. Because $V_2 \in \mathcal{V}$, $V_2 \cup \Theta \vdash \bot$. And because $R \in \mathcal{R}_\subseteq$, from the definition of $\mathcal{R}_\subseteq$, $R \cup \Theta \nvdash \bot$. So $V_2 \not\subseteq R$. Therefore there is a $\gamma \in V_2$ such that $\gamma \notin R$. By hypothesis, $\phi \in R$, so $\gamma \neq \phi$. And because $V_2 \setminus \{\phi\} \subseteq V_1$, $\gamma \in V_1$.
Then because $\phi' \notin \bigcup \text{hit}(\mathcal{V}, \{\phi\})$, $\phi' \notin V_2$, and therefore $\gamma \neq \phi'$ as well. So for any $V_1 \in \text{hit}(\mathcal{V}, \{\phi'\})$, there is a $\gamma \in V_1$ such that $\gamma \neq \phi'$ and $\gamma \notin R$. As a consequence, there is no $V_1 \in \text{hit}(\mathcal{V}, \{\phi'\})$ such that $V_1 \setminus \{\phi'\} \subseteq R$, and from the contraposition of lemma 8.8.5.2, and the fact that $R \cup \Theta \nvdash \bot$, $R \cup \{\phi'\} \cup \Theta \nvdash \bot$ must hold. Then from the definition of $\mathcal{R}_\subseteq$, if $R \cup \{\phi'\} \cup \Theta \nvdash \bot$, then $\phi' \in R$ must hold. □

8.8.5.2 Proposition 8.5.0.3

Proposition. $f_1(n) = O(n^3)$

Proof. Proposition 8.5.0.2 is reproduced here for readability:
Let $\phi \in \bigcup \mathcal{V}$.
If $\{\phi\} \in \mathcal{V}$, then $\bigcap \mathcal{S}_\phi = \emptyset$.
Otherwise $\bigcap \mathcal{S}_\phi = K_s \cup \{\phi\} \cup \Theta \cup$
$\{\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\}) \mid \forall V_1 \in \text{hit}(\mathcal{V}, \{\phi'\}), \exists V_2 \in \text{hit}(\mathcal{V}, \{\phi\}) : V_2 \setminus \{\phi\} \subseteq V_1\}$

As a reminder, f_1 expresses the cost of computing $\bigcap \mathcal{S}_\phi$ out of $\mathcal{V}$ and as a function of $n = \sum_{V \in \mathcal{V}} |V|$.

For the first condition, $\{\phi\} \in \mathcal{V}$ can be verified in $O(n)$.

For the second condition, let $\Gamma = \{\phi' \in \bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\}) \mid \forall V_1 \in \text{hit}(\mathcal{V}, \{\phi'\}), \exists V_2 \in \text{hit}(\mathcal{V}, \{\phi\}) : V_2 \setminus \{\phi\} \subseteq V_1\}$. Then $\Gamma \subseteq \bigcup \mathcal{V}$. And because $K_s \cup \Theta$ is the complement of $\bigcup \mathcal{V}$ in $K \cup \Theta$, $(K_s \cup \Theta) \cap \Gamma = \emptyset$, such that if Γ is known, the cost of computing $K_s \cup \{\phi\} \cup \Theta \cup \Gamma$ expressed as a function of n is constant.

So we can focus on the computation of Γ. First, two lookup tables can be built in $O(n)$, from each $V \in \mathcal{V}$ to itself, and from each $\phi_1 \in \bigcup \mathcal{V}$ to $\text{hit}(\mathcal{V}, \{\phi_1\})$. Then the number of axioms in $\bigcup \mathcal{V} \setminus \bigcup \text{hit}(\mathcal{V}, \{\phi\})$ is bounded by $|\bigcup \mathcal{V}| \leq n$, and each of $|\text{hit}(\mathcal{V}, \{\phi'\})|$ and $|\text{hit}(\mathcal{V}, \{\phi\}|$ is bounded by $|\mathcal{V}| \leq n$, such that the number of set inclusion verifications of the form $V_2 \setminus \{\phi\} \subseteq V_1$ to perform is bounded by $n \cdot n^2$. Finally, $|V_1| + |V_2| \leq n$, so the cost of such a set inclusion verification is in $O(n \log n)$. So $f_1(n) = O(O(n) + 2O(n) + n(n^2 n \log n)) = O(n^3)$. □

8.8.5.3 Proposition 8.5.0.4

Proposition. Let $\phi \in \bigcup \mathcal{V}$.
Then $\bigcap S_{\setminus \phi} = K_s \cup \Theta \ \cup \{\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\}) \mid$
if $\text{hitDiff}(\mathcal{V}, \phi', \phi) \neq \emptyset$ and $\text{hitDiff}(\mathcal{V}, \phi, \phi') \neq \emptyset$,
then $\forall V_1 \in \text{hitDiff}(\mathcal{V}, \phi', \phi), \forall V_2 \in \text{hitDiff}(\mathcal{V}, \phi, \phi') : ((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta \vdash \bot \}$

8.8.5.3.1 Left inclusion ($\subseteq$).

By contraposition.
Let us assume that ϕ' is not an element of the set on the right hand side of the equality. Then we need to show that if $S_{\setminus \phi} \neq \emptyset$, if there is a $\Delta \in S_{\setminus \phi}$ such that $\phi' \notin \Delta$, or equivalently that there is an $R \in \mathcal{R}_{\subseteq}$ such that $\phi \notin R$ and $\phi' \notin R$.

By hypothesis, $\phi' \notin K_s \cup \Theta$.

The first possible case is the one one where $\phi' \notin \bigcup \mathcal{V} \setminus \{\phi\}$. The first possible subcase is $\phi' \notin \bigcup \mathcal{V}$. But by hypothesis, $\phi' \notin K_s \cup \Theta$. So if $\phi' \notin \bigcup \mathcal{V}$ either, we have

$\phi' \notin K_s \cup \bigcup \mathcal{V} \cup \Theta = K \cup \Theta$, i.e. ϕ' is not in the input base. But for any $R \in S_{\setminus\phi}$, $R \subseteq K \cup \Theta$, so $\phi' \notin R$ trivially holds. The second possible subcase is $\phi' = \phi$. If there is an $R \in S_{\setminus\phi}$, then from the definition of $S_{\setminus\phi}$, $\phi \notin R$, so $\phi' \notin R$ trivially holds.

The second possible case is the one where $\phi' \in \bigcup \mathcal{V} \setminus \{\phi\}$, and there is a $V_1 \in \text{hitDiff}(\mathcal{V}, \phi', \phi)$ and a $V_2 \in \text{hitDiff}(\mathcal{V}, \phi, \phi')$ such that if $F \doteq V_1 \cup V_2 \setminus \{\phi, \phi'\}$, then $F \cup \Theta \nvdash \bot$. In this case, $V_1 \subseteq F \cup \{\phi'\}$, and because $V_1 \in \mathcal{V}$, $V_1 \cup \Theta \vdash \bot$, so by monotonicity, $F \cup \{\phi'\} \cup \Theta \vdash \bot$. Similarly, because $V_2 \subseteq F \cup \{\phi\}$, $F \cup \{\phi\} \cup \Theta \vdash \bot$ as well. So from lemma 8.8.5.1, replacing A by F and B by $\{\phi, \phi'\}$, there is an $R \in \mathcal{R}_\subseteq$ such that $\phi \notin R$ and $\phi' \notin R$.

8.8.5.3.2 Right inclusion ($\supseteq$)

Let us assume that ϕ' is an element of the set on the right-hand side of the equality, We need to show that $\phi' \in \bigcap S_{\setminus\phi}$, i.e. assuming an $R \in \mathcal{R}_\subseteq$ such that $\phi \notin R$, we need to show that $\phi' \in R$.

If $\phi' \in K_s \cup \Theta$, then $\phi' \notin \bigcup \mathcal{V}$ from the definitions of K_s and $\mathcal{V}$, so $\text{hit}(\mathcal{V}, \{\phi'\}) = \emptyset$, and from the contraposition of lemma 8.8.5.2, there is no $C \subseteq K$ such that both $C \cup \Theta \nvdash \bot$ and $C \cup \{\phi'\} \cup \Theta \vdash \bot$ hold. In particular, because $R \subseteq K$ and $R \cup \Theta \nvdash \bot$, $R \cup \{\phi'\} \cup \Theta \nvdash \bot$ cannot hold. But from the definition of $\mathcal{R}_\subseteq$, if $\phi' \notin R$, then $R \cup \{\phi'\} \cup \Theta \vdash \bot$ must hold. So $\phi' \in R$.

The next case is the one where $\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\})$, but $\text{hitDiff}(\mathcal{V}, \phi', \phi) = \emptyset$, i.e. $\phi \in \bigcap(\text{hit}(\mathcal{V}, \phi'))$. Let $R \in \mathcal{R}_\subseteq$ such that $\phi \notin R$, and assume by contradiction that $\phi' \notin R$. Then because $R \in \mathcal{R}_\subseteq$, $R \cup \Theta \nvdash \bot$ and $R \cup \{\phi'\} \cup \Theta \vdash \bot$ both hold, so from lemma 8.8.5.2, there must be a $V \in \text{hit}(\mathcal{V}, \phi')$ such that $V \setminus \{\phi'\} \subseteq R$, but then because $\phi \in \bigcap(\text{hit}(\mathcal{V}, \phi'))$ and $\phi \neq \phi'$, it must be the case that $\phi \in (V \setminus \{\phi'\}))$, so we would have $\phi \in R$, which contradicts the hypothesis.

The following case is very similar : it is the one where $\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\})$, but

$\text{hitDiff}(\mathcal{V}, \phi, \phi') = \emptyset$, i.e. $\phi' \in \bigcap(\text{hit}(\mathcal{V}, \phi))$. Let $R \in \mathcal{R}_{\subseteq}$ such that $\phi \notin R$, and assume by contradiction that $\phi' \notin R$. Then because $R \in \mathcal{R}_{\subseteq}$, $R \cup \Theta \nvdash \bot$ and $R \cup \{\phi\} \cup \Theta \vdash \bot$ both hold, so from lemma 8.8.5.2, there must be a $V \in \text{hit}(\mathcal{V}, \phi)$ such that $V \setminus \{\phi\} \subseteq R$, but then because $\phi' \in \bigcap(\text{hit}(\mathcal{V}, \phi'))$ and $\phi \neq \phi'$, it must be the case that $\phi' \in (V \setminus \{\phi\}))$, so we would have $\phi' \in R$, which contradicts the hypothesis.

So one can focus on the last and more interesting case, where $\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\})$, $\text{hitDiff}(\mathcal{V}, \phi', \phi) \neq \emptyset$, and ϕ' is nonetheless an element of the set on the right hand side of the equality. Then $\text{hitDiff}(\mathcal{V}, \phi, \phi') \neq \emptyset$, and $\forall V_1 \in \text{hitDiff}(\mathcal{V}, \phi', \phi)$, $\forall V_2 \in \text{hitDiff}(\mathcal{V}, \phi, \phi') : ((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta \vdash \bot$. We have to show that ϕ' is also an element of the set on the left hand side of the equality, i.e. $\phi' \in \bigcap S_{\setminus\{\phi\}}$.

By contradiction, assume that there is an $R \in \mathcal{R}_{\subseteq}$ such that $\phi \notin R$, and that $\phi' \notin R$. Because $\phi \notin R$, and $R \in \mathcal{R}_{\subseteq}$, from the definition of $\mathcal{R}_{\subseteq}$, both $R \cup \Theta \vdash \bot$ and $R \cup \{\phi\} \cup \Theta \vdash \bot$ must hold. So from lemma 8.8.5.2, there is a $V_1 \in \text{hit}(\mathcal{V}, \{\phi\})$ such that $V_1 \setminus \{\phi\} \subseteq R$. But as $\phi' \notin R$ and $\phi' \neq \phi$, it must also be the case that $\phi' \notin V_1$, and so $V_1 \in \text{hitDiff}(\mathcal{V}, \phi, \phi')$.

Identically, because $\phi' \notin R$, there is a $V_2 \in \text{hit}(\mathcal{V}, \{\phi'\})$ such that $V_2 \setminus \{\phi'\} \subseteq R$. And as $\phi \notin R$ and $\phi' \neq \phi$, $V_2 \in \text{hitDiff}(\mathcal{V}, \phi', \phi)$.

So $((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \subseteq R$. But $((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta \vdash \bot$, so by monotonicity $R \cup \Theta \vdash \bot$, and thus, from the definition of $\mathcal{R}_{\subseteq}$, $R \notin \mathcal{R}_{\subseteq}$, which contradicts the hypothesis. □

8.8.5.4 Proposition 8.5.0.5

Proposition. $f_2(n) = O(n^4)$

Proof. Proposition 8.5.0.4 is reproduced here for readability:

Let $\phi \in \bigcup \mathcal{V}$.

Then $\bigcap S_{\setminus \phi} = K_s \cup \Theta \ \cup \{\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\}) \mid$

if $\text{hitDiff}(\mathcal{V}, \phi', \phi) \neq \emptyset$ and $\text{hitDiff}(\mathcal{V}, \phi, \phi') \neq \emptyset$,

then $\forall V_1 \in \text{hitDiff}(\mathcal{V}, \phi', \phi)$, $\forall V_2 \in \text{hitDiff}(\mathcal{V}, \phi, \phi')$:

$((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta \vdash \bot \}$

As a reminder, f_2 expresses the cost of computing $\bigcap \mathcal{S}_{\setminus \phi}$ given $\mathcal{V}$ and $K \cup \Theta$, as a function of $n = \sum_{V \in \mathcal{V}} |V|$.

Let $\Gamma = \{\phi' \in ((\bigcup \mathcal{V}) \setminus \{\phi\}) \mid$ if $\text{hitDiff}(\mathcal{V}, \phi', \phi) \neq \emptyset$ and $\text{hitDiff}(\mathcal{V}, \phi, \phi') \neq \emptyset$, then $\forall V_1 \in \text{hitDiff}(\mathcal{V}, \phi', \phi)$, $\forall V_2 \in \text{hitDiff}(\mathcal{V}, \phi, \phi') : \text{Cn}(((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta) \cap \Psi \neq \emptyset \}$.

Then $\Gamma \subseteq \bigcup \mathcal{V}$. And because $K_s \cup \Theta$ is the complement of $\bigcup \mathcal{V}$ in $K \cup \Theta$, $(K_s \cup \Theta) \cap \Gamma = \emptyset$, such that if Γ is known, the cost of computing $K_s \cup \Theta \cup \Gamma$ expressed as a function of n is constant.

So we can focus on the computation of Γ. First, two lookup tables can be build in $O(n)$, from (a key for) each $V \in \mathcal{V}$ to itself, and from each $\phi_1 \in \bigcup \mathcal{V}$ to (a set of keys for each $V \in$) $\text{hit}(\mathcal{V}, \{\phi_1\})$.

Then the number of axioms in $(\bigcup \mathcal{V}) \setminus \{\phi\}$ is $|\bigcup \mathcal{V}| - 1 \leq n$. For each $\phi' \in (\bigcup \mathcal{V}) \setminus \{\phi\}$, let $m_1 = |\text{hit}(\mathcal{V}, \{\phi\})| + |\text{hit}(\mathcal{V}, \{\phi'\})|$. Then $m_1 \leq 2n$, and computing $\text{hitDiff}(\mathcal{V}, \phi', \phi)$ is in $O(m_1 \log m_1)$, and similarly for $\text{hitDiff}(\mathcal{V}, \phi, \phi')$.

Finally, $|\text{hitDiff}(\mathcal{V}, \phi', \phi)| \cdot |\text{hitDiff}(\mathcal{V}, \phi', \phi)|$ is (loosely) bounded by $|\mathcal{V}|^2$, and so is the number of pairs $< V_1, V_2 >$. Then for each of these pairs, verifying whether $\text{Cn}(((V_1 \cup V_2) \setminus \{\phi, \phi'\}) \cup \Theta) \cap \Psi \neq \emptyset\}$ can be reduced to verifying for each $V_3 \in (\mathcal{V} \setminus \{V_1, V_2\})$ whether $V_3 \subseteq (V_1 \cup V_2) \setminus \{\phi, \phi'\})$. The number of candidates for V_3 is $|\mathcal{V}| - 2$, so the number of such verifications is bounded by $|\mathcal{V}|^2 \cdot (|\mathcal{V}| - 2) \leq |\mathcal{V}|^3 \leq n^3$. Now let $m_2 = |V_1| \cup |V_2| - 2 + |V_3|$. Then $m_2 \leq 2n$, And each of these verifications is in $O(m_2 \log m_2)$.

So $f_2(n) = O(O(1) + 2O(n) + n(2O(n \log n) + n^3 O(n \log n)) = O(n^4)$ $\square$

Chapter 9

Conclusion

This thesis investigates different strategies in order to detect and repair violations of common sense in OWL knowledge bases (KBs). When it grows in size, an OWL KB is likely to contain logical statements which may make sense individually, but convey intuitively absurd information when considered together, even if the KB is logically consistent/coherent. Such violations of common sense greatly affect the reliability of the KB, in particular the knowledge which may be inferred from it. This phenomenon is exacerbated when the KB aggregates knowledge from multiple sources, which is one of the core mechanisms of the Semantic Web project.

The problem in itself is not particularly new: nonsense due to multiple meanings of a same predicate has been identified as a practical limitation of logic-based knowledge representation systems since at least the expert systems era. But to our knowledge, no effective strategy has been developed yet which allows for detecting and solving such cases in an automated fashion. In particular, these errors often remain unnoticed by state-of-the KB debugging tools like the ones presented in Chapter 3 Section 3.4.

The main topic under investigation in this dissertation is the incorporation of automatically gathered linguistic evidence to a debugging process, in order to detect and repair such errors, which is in itself an original proposal. Indeed, the possible use of Natural Language Processing (NLP) techniques for KB engineering has mostly been studied in the fields of information extraction or ontology learning/population, where the task consists in extending an existing KB, or creating one from scratch. But these tasks fundamentally differ from debugging. In particular, ontology learning/population extends the signature of the KB under construction with additional individuals and/or predicates identified in natural language texts, whereas in a debugging setting, the signature of the KB is known in advance. Therefore adapting such techniques to the case of debugging is not straightforward, and for several of them, probably not relevant.

9.1 Main contributions

9.1.1 Linguistic evidence

Chapter 4 proposes to exploit an assumption made by some works on ontology population, namely the fact that individuals which instantiate the same concepts according to a KB also tend to have similar linguistic behaviors. The choice is made to rely on distributional representations of the labels associated with these individuals in the KB, obtained from a collection of automatically retrieved web pages.

If Q is a candidate output KB of the debugging process, a so-called *plausibility score* is proposed which, for some syntactically defined finite subset Ψ_Q of the consequences of Q, and for each $\psi \in \Psi_Q$, estimates to what extent ψ is likely to hold if the rest of Ψ_Q does, based on distributional representations. Intuitively, the higher these

plausibility scores, the more Q is in line with the above assumption, i.e. individuals which instantiate the same concepts according to Q tend to have similar linguistic behavior.

Then if $\mathcal{Q}$ is the family of candidate output bases of a debugging process, Section 4.2.5 defines four alternative preference relations over $\mathcal{Q}$ based on these plausibility scores.

9.1.2 Integration to a KB debugging process

9.1.2.1 Consistent input KB

Chapter 6 investigates the integration of the four preference relations just mentioned to an actual debugging process, in the case where the input KB K is consistent, and where debugging consists in selecting a subset of K for removal.

If a single erroneous axiom needs to be identified within K, the integration is straightforward.

But if an arbitrary set of erroneous axioms needs to be identified within K instead, i.e. if the family $\mathcal{Q}$ of candidate output bases is 2^K, it is shown that no obvious algorithmic solution allows for computing the optimal elements of $\mathcal{Q}$.

Section 6.3.2 provides an (arguably complex) procedure which yields these preferred subbases for one of the four preference relations, but may be costly in practice. In particular, it is shown that the problem is as hard as computing for each $\psi \in \Psi_K$ the family of all maximal subbases of K which do not entail ψ.

9.1.2.2 Inconsistent/incoherent input KB

Linguistic evidence as formalized in in Chapter 4 presents an interesting complementarity with so-called syntax-based KB contraction/revision algorithms developed for

Description Logics, usually for inconsistent/incoherent KBs. These algorithm select (a) minimal set(s) of axioms to discard from the input KB K in order to restore consistency/coherence, or equivalently, they select the complement(s) of these discarded axioms as (an) output base(s). In practice, these approaches suffer from the number of candidate output bases, when no other criterion than maximality is available to operate the selection. In this case, the compliance of a candidate subbase Q to the abovementioned assumption (namely that individuals which instantiate the same concepts tend to have similar linguistic behavior) offers an interesting additional criterion to rank candidate output bases.

This proposal remains theoretical though, because for most inputs, syntax-based KB contraction/revision algorithms also suffer from computational cost issues, if all candidate output bases are to be computed. For this reason, Chapter 8 provides a *prioritized* syntax-based contraction/revision algorithm. Provided the family $\mathcal{J}$ of so-called *justifications* for the inconsistency/incoherence is known, and given a preference relation over the axioms of the input KB, the algorithm consists in prioritizing the removal of least preferred axioms until consistency/coherence is reached, without the need to compute all candidate output bases.

An original proposal is also made in order to obtain this preference relation over the axioms of the input KB, in the form of two bases which, for each candidate axiom ϕ for removal, represent what part of the input KB would be necessarily retained if ϕ was respectively retained or discarded, and can be computed in time polynomial in $\sum_{J \in \mathcal{J}} |J|$.

These two proposals can be considered as independent contributions to the field of syntax-based contraction/revision, in that their application is not restricted to the case where linguistic evidence is incorporated to the process.

9.1.3 Experiments

Different evaluations are presented throughout this dissertation, some of which focus on local contributions, other ones evaluating more general debugging strategies. In addition, two alternative types of evaluation protocols were implemented. The first one relies on real datasets, and the evaluation consists in manually reviewing the axioms automatically identified as erroneous (or potentially erroneous in some cases) by a system. But because this manual evaluation may be held as subjective (and is costly), an alternative protocol was designed, which is based on automatically degraded datasets, obtained by extending an input KB with axioms randomly generated out of its signature (the procedure is described in Chapter 5), the underlying assumption being that random axioms are very likely to be absurd wrt to the rest of the KB. Then the evaluation consists in verifying whether a system was able to retrieve these random axioms (or sometime new consequences due to the addition of these axioms) on a linguistic basis.

Almost all evaluations presented in this thesis produced significant (positive) results.

9.1.3.1 Plausibility

Section 6.1 describes an independent evaluation of the plausibility score defined in Chapter 4, although plausibility scores by themselves do not provide an effective debugging strategy. 200 automatically degraded KBs were used for this evaluation, which consisted in verifying whether the additional consequences obtained after extending the KB with random axioms were also the ones with lowest plausibility.

9.1.3.2 Trimming a single axiom

Section 6.2 presents an evaluation whose objective was to automatically spot a single randomly generated axiom in a KB. In other words, if K is the input KB before degradation and ϕ the randomly generated axiom, the goal of this evaluation was to identify K as the preferred immediate subbase of $K \cup \{\phi\}$, on a linguistic basis.

9.1.3.3 Trimming multiple axioms

Section 6.3.2.1 evaluates the iterated removal of the most unlikely axiom of a KB on a linguistic basis, with up to 20 iterations, i.e. 20 discarded axioms. This strategy was evaluated manually with a real dataset, and automatically with a degraded one, with 20 randomly generated axioms to retrieve.

9.1.3.4 Ontological analysis

Section 7.4 evaluates the manual extension of a consistent input KB with a foundational ontology, based on the strategy introduced in Section 7.3, which does not depend on linguistic evidence. This strategy consists in performing a (fast and shallow) ontological analysis of some elements of the signature of a consistent input KB, and was experimented as a complementary debugging step, in order to yield an inconsistent KB, which then may serve as the input of a syntax-based KB revision algorithm.

The evaluation shows that this type of shallow ontological analysis may indeed identify erroneous axioms as involved in the inconsistency, such that these axioms are among candidates for removal during the syntax-based revision phase.

9.1.3.5 Syntax-based revision guided by linguistic evidence

Section 8.6 evaluates some of the proposals made in Chapter 8, in the specific case where the preference relation over axioms of the inconsistent KB is obtained from linguistic evidence, and taking as input two datasets manually extended with a foundational ontology, following the strategy described in Chapter 7.

9.2 Cohesion

The cohesion of this work is primarily thematic, dictated by a global objective, which is the detection and repair of nonsense within an OWL KB.

But at a lower level, the specific problems addressed throughout this thesis are largely independent. For instance, the manual KB extension strategy described in Chapter 7 is largely disconnected from the proposals made in Chapters 6 and 8, and Chapter 8 in itself can be read as an independent contribution to syntax-based contraction/revision, without any relation to NLP. In addition, as already mentioned, the sequencing of Chapters 6 to 8 is not meant to reflect a unique debugging process.

So this thesis should primarily be viewed as a pluridisciplinary series of more or less correlated contributions, only guided by the same initial motivation. In particular, as explained below, some of these contributions may find applications in very different contexts.

9.3 Scope and interpretation

9.3.1 Non-exhaustiveness

As is often the case for pluridisciplinary works, the topics and problematics are largely new, and this thesis is by no means an exhaustive investigation of the applicability of NLP to KB debugging. In particular, the focus on distributional similarity and individuals was guided by robustness, and discussed in depth in Chapter 4 Sections 4.1 and 4.2.1:

But a wide array of other NLP techniques may be investigated, some of which are mentioned in Chapter 4, and no claim is made here as to whether these solutions are more efficient than the ones investigated in this work.

9.3.2 Interpretations of the empirical results

Because of the relative novelty of the problem, no existing benchmark was available to evaluate the different proposals made throughout this thesis, neither was a standard evaluation protocol.

Therefore one may rightfully wonder to what extent the relatively good results obtained in these experiments are biased by ad hoc evaluation conditions or datasets, and more generally, whether the evaluated debugging strategies may successfully be applied to arbitrary input KBs.

9.3.2.1 Dataset selection

None of the datasets described in Chapter 5 was constituted to yield artificially good results. The target application scenario is a set of up to a few thousand axioms,

prototypically extracted from the LOD Cloud. Subsets of DBpedia seem like good candidates, because DBpedia is the *de facto* core of the LOD Cloud. The (arguably complex) DBpedia subset extraction procedure described in Chapter 5 Section 5.1.3 was only designed to work with more realistic sets of statements as input, based on the assumption that if a subset K of a large and domain independent KB is extracted for a given application, then K will probably present a relative topical cohesion, as discussed in Section 5.1.3.

The two datasets K_{STLab} and $K_{fisheries}$ used for automatic degradation on the other hand were chosen by default, for practical reasons. It is indeed difficult to find small KBs which both are of a high quality and include named individuals with linguistic labels, most reference KBs produced by academic research being TBoxes.

9.3.2.2 Parameters

As may have appeared to the attentive reader, the number of tunable parameters for each experiment was generally high. A good illustration is the evaluation focused on linguistic evidence in Chapter 6, where the number of provided results is already important, due to the different combinations of tested configurations (types of linguistic contexts and weighting methods). But several other parameters could also have been tested with different values, for instance the size of these linguistic contexts, the number of web pages retrieved for each named individual, ... Thresholds may also have been added (as is sometimes made in distributional semantics) for the minimal number of target words a linguistic context appears with. The number of tunable parameters grows with the complexity of the overall debugging procedures in the later chapters, in particular for the evaluation described in Chapter 8, and exploring all possible combinations was not practically feasible in the context of this

thesis.

So some parameters were indeed tuned to obtain good results, and the values retained for these parameters are provided with the description of each experiment. But the possibility remains that other values for these parameters may be more efficient for other inputs, i.e. nothing guarantees that these parameters do not depend on the specific datasets used for the experiments.

Therefore these results should be interpreted cautiously. When they are statistically significant (and they often are), this should be viewed as an illustration that the underlying intuitions are relevant, but some adjustments may still be needed to obtain comparable results on other datasets.

9.4 Continuation

Although KB debugging guided by linguistic evidence was the main motivation behind the contributions made in this thesis, several of them may actually find applications in other contexts. In particular:

- Plausibility scores, when aggregated, evaluate to what extent a set of formulas verifies the above linguistic assumption, namely that individuals which instantiate the same concepts tend to have similar linguistic behavior. Therefore they may find an application in some engineering contexts where two sets of formulas need to be compared, or even in an ontology learning scenario, when some candidate set of extracted formulas needs to be filtered, as an alternative for instance to the heuristic developed by [NVF11].

- The problem addressed by the algorithm proposed in Section **??** can be formulated in more generic terms, and find applications in other knowledge engineer-

ing scenarios: given a consistent KB K, and a finite set Ψ_K of consequences of K with a preference relation $\preceq$ over Ψ_K, find the optimal subbase(s) of K according to the standard lexicographic ordering defined by $\preceq$ over all $K' \cap \Psi_K$ such that $K' \subseteq K$.

- As mentioned above, the proposals made in Chapter 8 can be viewed as independent contributions to the field of syntax-based revision/contraction.

9.4.1 Scalability

For very expressive DLs, most of the debugging procedures proposed in this thesis are intractable, because they rely on the derivation of some consequences of candidate output bases. But even in DLs for which entailment is tractable, the cost of the execution of several of the algorithms proposed in this thesis is exponential in the worst case in the size of the KB.

At most, the datasets used for the evaluations count a few thousand axioms, and the execution of some algorithms turned out to be empirically costly. Therefore it can be safely assumed that applying these algorithms to very large KBs is not a realistic scenario, and approximations should be developed instead. But even if the implementation differs, results presented in this thesis tend to indicate that the intuitions underlying the proposals made in Chapter 4 (for the integration of linguistic evidence to a KB debugging process) are promising.

Appendix A

Relevance postulate for revision in $\mathcal{ALCHO}$

This appendix focuses on belief set revision, as opposed to the proposals made in Chapter 8, which focused on belief bases. The observation made in this appendix partly rely on results provided in Chapter 6 of [Rib13] (and the work presented in [Rib13] is itself partly based on [Flo06]). Therefore the notation adopted in this appendix is closer to the one used in [Rib13]. In particular, a *logic* will designate a pair $< \mathcal{L}, \mathrm{Cn} >$, with $\mathcal{L}$ the (syntactically defined) language of the logic, and $\mathrm{Cn} : 2^{\mathcal{L}} \mapsto 2^{\mathcal{L}}$ a consequence operator. A logic $< \mathcal{L}, \mathrm{Cn} >$ is said to be *Tarskian* iff for any $\Gamma, \Gamma_1, \Gamma_2 \subseteq \mathcal{L}$, the three following hold:

$\Gamma \subseteq \mathrm{Cn}(\Gamma)$	((Tarskian) inclusion)
$\mathrm{Cn}(\Gamma) = \mathrm{Cn}(\mathrm{Cn}((\Gamma))$	(indempotence)
if $\Gamma_1 \subseteq \Gamma_2$, then $\mathrm{Cn}(\Gamma_1) \subseteq \mathrm{Cn}(\Gamma_2)$	(monotonicity)

The term "(Tarskian) inclusion" is used here in order to avoid possible confusions with the "inclusion" postulate for revision described below (Section A.1.4)

The focus will be put on the DL $\mathcal{ALCHO}$, which is $\mathcal{ALC}$ (introduced in Chapter 2 Section 2.3.2) extended with nominals and role subsumption. The syntax of $\mathcal{ALCHO}$ is defined by the grammar of figure A-1 below (Section A.1.6), and its (standard model-theoretic) semantic is the one given in Chapter 2 Section 2.3.4 for the DL $\mathcal{SROIQ}$, but restricted to the syntactic subset of $\mathcal{SROIQ}$ defined by the grammar of figure A-1.

The main topic of this appendix is the minimal change postulate for revision (not for contraction) proposed by [Rib13] for some Tarskian logics which are not *AGM compliant* according to the authors, i.e. for which no contraction and/or revision operator exists which satisfies all AGM postulates for contraction and/or revision, because these logics do not verify some properties, such as (finite) complementation or *decomposability* (see below), which are verified by classical propositional logic (among other logics).

$\mathcal{ALCHO}$ is interesting because it does not verify decomposability, as illustrated by example 3.6.1 in Chapter 3 Section 3.6.1.1. The formal definition of decomposability is not needed here (the reader is referred to [Flo06]). Instead, only the following sufficient condition for a logic to be non-decomposable will be reproduced from [Flo06]: if $< \mathcal{L}, \mathrm{Cn} >$ is a logic, for any $\Gamma \subseteq \mathcal{L}$, let $\mathrm{Cn}_{strict}(\Gamma)$ designate the set of strict consequences of Γ in $< \mathcal{L}, \mathrm{Cn} >$, i.e. $\mathrm{Cn}_{strict}(\Gamma) = \{\psi \in \mathcal{L} \mid \mathrm{Cn}(\{\psi\}) \subset \mathrm{Cn}(\Gamma)\}$. If there is an $\Omega \subseteq \mathcal{L}$ such that $\mathrm{Cn}(\mathrm{Cn}_{strict}(\Omega)) \neq \mathrm{Cn}(\Omega)$, then $< \mathcal{L}, \mathrm{Cn} >$ is not decomposable. It was shown in [Flo06] that a logic which does not verify decomposability is not *AGM compliant*, i.e. there is no contraction operator in this logic which satisfies all 6 first (generalized) AGM postulates for contraction (introduced in Chapter 3 Section 3.6.1.1). If $< \mathcal{L}, \mathrm{Cn} >= \mathcal{ALCHO}$, and if R and S are two DL atomic roles,

then $\mathrm{Cn}(\mathrm{Cn}_{strict}(\{R \sqsubseteq S\})) \neq \mathrm{Cn}(\{R \sqsubseteq S\})$ holds, such that $\mathcal{ALCHO}$ is not decomposable, and therefore it is not AGM compliant. $\mathcal{ALCHO}$ verifies a weaker property though, which will be useful in the proofs below (specifically, in the proof of lemma A.4.1.1): for all $\Gamma_1, \Gamma_2 \subseteq \mathcal{L}$, if $\mathrm{Cn}(\Gamma_1 \cup \Gamma_2) = \mathcal{L}$, then $\mathrm{Cn}(\mathrm{Cn}_{strict}(\Gamma_1) \cup \Gamma_2) = \mathcal{L}$.

As an alternative to recovery, for contraction still, [Rib13] proposed to use the *relevance* postulate for contraction, also defined in Chapter 3 Section 3.6.1.1, which, as shown by [Han91], is equivalent to recovery in AGM compliant logics, but is applicable to a larger class of logics, called *relevance compliant* by [Rib13], which includes non-decomposable logics, such as Horn Logic, intuitionistic logic, and most DLs (among which $\mathcal{ALCHO}$).

Dealing with the recovery postulate is not the only generalization of the AGM framework proposed by [Flo06] and [Rib13]. In particular, some of the logics already mentioned are not closed under disjunction and/or conjunction, or are not (finitely) complemented (see Section A.1.2). Therefore the postulates which will be used here are not exactly the ones defined in [AGM85], but these postulates generalized to sets of formulas, and to logics where the complement of a (finitely representable set of) formula(s) does not always exist. As explained in Chapter 3 Section 3.6.1.1, these generalizations are more straightforward than the generalization of recovery (to relevance), and will not be explained in details here (the reader is referred to [Rib13] instead).

The notion of AGM compliance, as defined by [Flo06], focuses on contraction, not on revision. This may be legitimated by the fact that there is no minimal change postulate for revision in the AGM framework, or more exactly, as explained in Chapter 3 Section 3.6.1.1, that the recovery postulate for revision is redundant in the presence of the other basic postulates for revision. As a consequence, the AGM framework alone does not provide a minimal information loss principle in the case

of revision. In particular, as explained in Chapter 3 Section 3.6.1.1, the revision operator which returns $K + \Theta$ if $K + \Theta$ is consistent, and $\mathrm{Cn}(\Theta)$ otherwise, satisfies all basic postulates for revision,[1] and therefore it also satisfies recovery for revision.

For relevance compliant but not AGM compliant logics though, [Rib13] defines a (generalized) relevance postulate for revision. The question investigated in this appendix is the conditions in which this postulate is redundant with the (generalized) basic postulates for revision defined by [Rib13], just like recovery for revision is redundant with the basic postulates for revision in the AGM framework. In particular, a property (Property A.2.1.1) is provided in Section A.2.1 which, if verified by a Tarskian logic, is sufficient for the relevance postulate for revision to be redundant with the (generalized) closure, success and vacuity postulates for revision. No claim is made though as to whether this property is also necessary, i.e. it may be too strong to characterize exactly this phenomenon. The implications for Tarskian logics verifying this property may be similar to the ones for AGM compliant logics, namely that an additional notion of minimal change (for instance model-based, like the ones introduced in Chapter 3 Section 3.6.1.2) is needed to guarantee a form of minimal information loss.

It is then shown in Section A.2.2 that $\mathcal{ALCHO}$ satisfies this property, such that the relevance postulate for revision is redundant in $\mathcal{ALCHO}$ with the basic postulates for revision. An interesting continuation of this work would be to determine whether or not this observation also holds for some more expressive DLs.

[1] as well as the two additional Gärdenfors postulates for revision

A.1 Preliminaries

A.1.1 Typographical conventions

A difference between the notation adopted in this appendix and the conventions used in the belief change literature is that no set of formulas will be assumed to be closed deductively, unless explicitly stated. In particular, K will denote a subset of $\mathcal{L}$, not necessarily closed deductively, and the closure of K will be explicitly designated as $\mathrm{Cn}(K)$. This choice is made in order to be consistent with the usage of other capital Greek letters in this appendix ($\Theta, \Gamma, \Omega, \ldots$) which all designate (not necessarily closed) sets of formulas. $K * \Theta$ on the other hand is assumed to be closed deductively, but only because this is explicitly required by the closure postulate below (Section A.1.4).

Another specific typographical convention is also used in what follows. For readability, if $\Gamma \subseteq \mathcal{L}$, then $\overline{\Gamma}$ will generally be used *as a variable name* in order to designate another subset of $\mathcal{L}$ such that $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}) = \mathcal{L}$.[2] It is important to understand that the "overline" symbol in $\overline{\Gamma}$ is *not* an operator here, but only a typographical convention. So in what follows, from a formal point of view, "$\overline{\Gamma}$" may as well be designated with "Ω, "Γ'" or "Γ_2", i.e. the "overline" symbol in $\overline{\Gamma}$ has no formal meaning, and is only a notational facility, because some of the propositions or proofs below may be difficult to read otherwise.

[2] If $\gamma \in \mathcal{L}$, then $\overline{\gamma}$ will also be used, *as a variable name* still, in order to designate another formula in $\mathcal{L}$ such that $\mathrm{Cn}(\{\gamma, \overline{\gamma}\}) = \mathcal{L}$.

A.1.2 Complementation

As noted in [FHP+06] or [Rib13], most DLs are not (finitely) complemented, i.e. the complement of a (finitely representable set of) formula(s) in a DL does not always exist. More formally, if $< \mathcal{L}, \mathrm{Cn} >$ is the (Tarskian) logic at hand, let $f(< \mathcal{L}, \mathrm{Cn} >)$ be the family of finitely representable subsets of $\mathcal{L}$ according to $< \mathcal{L}, \mathrm{Cn} >$, defined by:

Definition A.1.2.1. $f(< \mathcal{L}, \mathrm{Cn} >) = \{\Gamma \subseteq \mathcal{L} \mid$ there is a finite $\Gamma' \subseteq \mathcal{L}$ such that $\mathrm{Cn}(\Gamma) = \mathrm{Cn}(\Gamma')\}$

If $\Gamma \in f(< \mathcal{L}, \mathrm{Cn} >)$, then $\overline{\Gamma} \subseteq \mathcal{L}$ is a complement of Γ iff $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}) = \mathcal{L}$ and $\mathrm{Cn}(\Gamma) \cap \mathrm{Cn}(\overline{\Gamma}) = \mathrm{Cn}(\emptyset)$.[3] Because there is no constraint here on the syntactic form of $\overline{\Gamma}$, there may be several complements for a same $\Gamma \in f(< \mathcal{L}, \mathrm{Cn} >)$. But as noted by [Rib13], there may also be several complements of Γ in $< \mathcal{L}, \mathrm{Cn} >$ modulo equivalence.

An arguably more common technical difficulty though is the case where there is no finitely representable complement of Γ. For instance, in FOL, let $\Gamma_1 = \{\forall x : A(x)\}$, with A a unary FOL predicate. Then any complements of Γ_1 in FOL is equivalent to $\overline{\Gamma_1} = \{\exists x : \neg A(x)\}$. And under the standard model-theoretic semantic for FOL and DLs defined in Chapter 3 Section 2.3.4, a model-preserving translation of Γ_1 into the DL $\mathcal{ALC}$ is $\Gamma_2 = \{\top \sqsubseteq A\}$, with A a DL atomic concept. But there is no finite set of formulas in $\mathcal{ALC}$ which is a model-preserving translation of $\overline{\Gamma_1}$. And there is no finite set $\overline{\Gamma_2}$ of formulas in $\mathcal{ALC}$ which satisfies both of the properties above either, i.e. which satisfies $\mathrm{Cn}(\Gamma_2 \cup \overline{\Gamma_2}) = \mathcal{L}$ and $\mathrm{Cn}(\Gamma_2) \cap \mathrm{Cn}(\overline{\Gamma_2}) = \mathrm{Cn}(\emptyset)$. In

[3] For the second condition, $\mathrm{Cn}(\mathrm{Cn}(\Gamma) \cap \mathrm{Cn}(\overline{\Gamma})) = \mathrm{Cn}(\emptyset)$ is arguably a more intuitive notation, but because the intersection of two belief sets in a Tarskian logic is a belief set, $\mathrm{Cn}(\mathrm{Cn}(\Gamma) \cap \mathrm{Cn}(\overline{\Gamma}))$ and $\mathrm{Cn}(\Gamma) \cap \mathrm{Cn}(\overline{\Gamma})$ both designate the same set of formulas, and the shorter notation is preferred here.

particular, $\Gamma_3 = \{\top \sqsubseteq \neg A\}$ is not a complement of Γ_2. It satisfies $\mathrm{Cn}(\Gamma_2 \cup \Gamma_3) = \mathcal{L}$, but not $\mathrm{Cn}(\Gamma_2) \cap \mathrm{Cn}(\Gamma_3) = \emptyset$. For instance, the formula $\psi = \top \sqsubseteq A \sqcup \forall R.\neg A$, is a consequence of both Γ_2 and Γ_3, i.e. $\psi \in (\mathrm{Cn}(\Gamma_2) \cap \mathrm{Cn}(\Gamma_3))$, but ψ is not a tautology, i.e. $\psi \notin \mathrm{Cn}(\emptyset)$. To see this, consider the interpretation $\mathcal{I} =< \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$ such that the domain of the interpretation function $.^{\mathcal{I}}$ is $\{A, R\}$, with A a DL atomic concept and R a DL atomic role, $\Delta^{\mathcal{I}} = \{\delta_1, \delta_2\}$, $A^{\mathcal{I}} = \{\delta_2\}$ and $R = \{(\delta_1, \delta_2)\}$. Then $\mathcal{I} \not\models \psi$, and therefore $\psi \notin \mathrm{Cn}(\emptyset)$.

This possible lack of a finitely representable complement for some $\Gamma \in f(< \mathcal{L}, \mathrm{Cn} >)$ is in particular the reason why the condition in the generalization of the vacuity postulate below is "if $K + \Theta = \mathcal{L}$", and not "if $\neg\theta \in \mathrm{Cn}(K)$ for all $\theta \in \Theta$", which would be more in line with its original formulation in the AGM framework (given in Chapter 3 Section 3.6.1.1), namely "if $\theta \in \mathrm{Cn}(K)$". Similarly, the definition of Property A.2.1.1 below relies on inconsistency instead of explicit negation.

In addition, as a notational shortcut, if $< \mathcal{L}, \mathrm{Cn} >$ is the logic at hand, the operator $! : 2^{\mathcal{L}} \mapsto 2^{2^{\mathcal{L}}}$ will be defined as follows:

Definition A.1.2.2. $!\Gamma = \{\overline{\Gamma} \subseteq \mathcal{L} \mid \mathrm{Cn}(\Gamma \cup \overline{\Gamma}) = \mathcal{L}\}$

A.1.3 Postulates

A.1.4 Generalized Postulates for Revision

The generalization of the basic AGM postulates for revision provided by [Rib13] characterize the revision of a set K of formulas by a set Θ of formulas (understood conjunctively), instead of the revision of K by a single formula θ in the AGM framework. This generalization also addresses the case of logics which may not be finitely complemented, which has just been discussed. The generalized postulates proposed

by [Rib13] are reproduced here:

(1) $K * \Theta = \mathrm{Cn}(K * \Theta)$ (closure)

(2) $\Theta \subseteq K * \Theta$ (success)

(3) $K * \Theta \subseteq K + \Theta$ (inclusion)

(4) if $K + \Theta \neq \mathcal{L}$, then $K + \Theta \subseteq K * \Theta$ (vacuity)

(5) if $\mathrm{Cn}(\Theta) \neq \mathcal{L}$, then $K * \Theta \neq \mathcal{L}$ (consistency)

(6) if for all $K' \subseteq \mathrm{Cn}(K)$, $K + \Theta_1 = \mathcal{L}$ iff $K + \Theta_2 = \mathcal{L}$,
then $\mathrm{Cn}(K) \cap (K * \Theta_1) = \mathrm{Cn}(K) \cap (K * \Theta_2)$ (extensionality)

A.1.5 Relevance postulate for revision

In addition to these postulates, [Rib13] introduces the following relevance postulate for revision, which can be viewed as an an adaptation to revision of the relevance postulate for contraction, via Harper's identity (see Chapter 3 Section 3.6.1.1):

If $\gamma \in \mathrm{Cn}(K) \setminus (K * \Theta)$, then there is a K' such that:

- $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K' \subseteq \mathrm{Cn}(K)$
- $\mathrm{Cn}(K' \cup \Theta) \neq \mathcal{L}$
- $\mathrm{Cn}(K' \cup \Theta \cup \{\gamma\}) = \mathcal{L}$ (relevance)

A.1.6 $\mathcal{ALCHO}$: Syntax

Figure A-1 gives the syntax of the DL $\mathcal{ALCHO}$, which is the DL $\mathcal{ALC}$ extended with role subsumption and nominals. As explained in Chapter 2 Section 2.3.7, although this grammar does not allow for the generation of ABox statements syntactically, it allows for the generation of TBox statements which are semantically equivalent,

ϕ	::=	$\phi_{TBox} \mid \phi_{RBox}$
ϕ_{TBox}	::=	$C \sqsubseteq C$
ϕ_{RBox}	::=	$\mathbf{R} \sqsubseteq \mathbf{R}$
C	::=	$\mathbf{A} \mid \top \mid \bot \mid \{N\} \mid \neg C \mid C \sqcup C \mid C \sqcap C \mid$ $\exists R.C \mid \forall R.C$
N	::=	$\mathbf{e} \mid \mathbf{e}, N$

Figure A-1: Syntax of $\mathcal{ALCHO}$

relying on nominals. For instance, $\{e\} \sqsubseteq C$ is equivalent to the ABox statement $C(e)$, with C a DL concept and e an individual, and $\{e_1\} \sqsubseteq \exists R.\{e_2\}$ is equivalent to $R(e_1, e_2)$, with e_1, e_2 two individuals and R a DL atomic role.

The standard model-theoretic semantic for $\mathcal{ALC}$ is the one for $\mathcal{SROIQ}$ given in Chapter 3 Section 2.3.4, but restricted to the language generated by the grammar of figure A-1.

A.2 Main propositions

A.2.1 Sufficient condition for the redundancy of relevance for revision

The following property may or may not be verified by a Tarskian logic $< \mathcal{L}, \mathrm{Cn} >$. But if it is verified, a revision operator in this logic which satisfies closure, vacuity and success also satisfies relevance for revision.

Property A.2.1.1. Let $\Gamma_1, \Gamma_2, \Gamma_3, \overline{\Gamma_3} \subseteq \mathcal{L}$, with $\Gamma_2 \not\subseteq \mathrm{Cn}(\Gamma_1)$ and $\mathrm{Cn}(\Gamma_3 \cup \overline{\Gamma_3}) = \mathcal{L}$. Then there are $\overline{\Gamma_2}, \Gamma_4 \subseteq \mathcal{L}$ such that:

a: $\mathrm{Cn}(\Gamma_2 \cup \overline{\Gamma_2}) = \mathcal{L}$

b: $\mathrm{Cn}(\Gamma_1 \cup \overline{\Gamma_2}) \neq \mathcal{L}$

c: $\mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_3}) = \mathcal{L}$

d: $\Gamma_4 \subseteq \mathrm{Cn}(\Gamma_3)$

e: $\overline{\Gamma_2} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$

Then the following shown to hold in Section A.3:

Proposition A.2.1.1. If a logic is Tarskian and satisfies property A.2.1.1, then a revision operator in this logic which verifies closure, success and vacuity also verifies relevance.

A.2.2 Application: the case of $\mathcal{ALCHO}$

The following is shown to hold in Section A.4:

Proposition A.2.2.1. $\mathcal{ALCHO}$ under standard model-theoretic semantic for DLs and without unique name assumption verifies property A.2.1.1.

Then because $\mathcal{ALCHO}$ is a Tarskian logic, the following proposition, already introduced in Chapter 3 Section 3.6.1.1, follows immediately from proposition A.2.1.1:

Proposition. If a revision operator in $\mathcal{ALCHO}$ satisfies closure, success, and vacuity, then it satisfies relevance.

A.3 Proof of proposition A.2.1.1

The relevance postulate for revision proposed in Chapter 6 of [Rib13] is reproduced here for readability:

If $\gamma \in \mathrm{Cn}(K) \setminus (K * \Theta)$, then there is a K' such that:

- $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K' \subseteq \mathrm{Cn}(K)$
- $\mathrm{Cn}(K' \cup \Theta) \neq \mathcal{L}$
- $\mathrm{Cn}(K' \cup \Theta \cup \{\gamma\}) = \mathcal{L}$ (relevance)

The following theorem will also be useful:

Theorem A.3.0.1. If $< \mathcal{L}, \mathrm{Cn} >$ is Tarskian and $X_1, X_2 \subseteq \mathcal{L}$, then $\mathrm{Cn}(\mathrm{Cn}(X_1) \cup X_2) = \mathrm{Cn}(X_1 \cup X_2)$

And this is the proposition to be proven:

Proposition. If a logic is Tarskian and satisfies property A.2.1.1, then a revision operator in this logic which verifies closure, success and vacuity also verifies relevance.

Proof. Let $< \mathcal{L}, \mathrm{Cn} >$ be a Tarskian logic, and let $*$ be a revision operator for $< \mathcal{L}, \mathrm{Cn} >$ satisfying closure, success and vacuity, with $K, \Theta \subseteq \mathcal{L}$ the two arguments of $*$.

Let us evacuate some limit cases first. If $K + \Theta \neq \mathcal{L}$ then by vacuity, $K + \Theta \subseteq K * \Theta$, and because $\mathrm{Cn}(K) \subseteq K + \Theta$, $\mathrm{Cn}(K) \subseteq K * \Theta$ must hold.

If $\mathrm{Cn}(\Theta) = \mathcal{L}$, by success, $K * \Theta = \mathcal{L}$, and so $\mathrm{Cn}(K) \subseteq K * \Theta$ holds as well.

So in both cases $\mathrm{Cn}(K) \setminus (K * \Theta) = \emptyset$, and relevance trivially holds.

Therefore we can focus on the case where $K + \Theta = \mathcal{L}$ (i.e $\mathrm{Cn}(K \cup \Theta) = \mathcal{L}$) and $\mathrm{Cn}(\Theta) \neq \mathcal{L}$.

We will show that if property A.2.1.1 is verified by $< \mathcal{L}, \mathrm{Cn} >$, then for any $\gamma \in \mathrm{Cn}(K) \setminus (K * \Theta)$, there is a set Ω of formulas such that $K' = \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Omega)$ verifies:

- $K' \subseteq \mathrm{Cn}(K)$

- $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K'$
- $\mathrm{Cn}(K' \cup \Theta) \neq \mathcal{L}$
- $\mathrm{Cn}(K' \cup \Theta \cup \{\gamma\}) = \mathcal{L}$

By hypothesis, $\gamma \notin K * \Theta$. And from the closure postulate, $\mathrm{Cn}(K * \alpha) = K * \Theta$, so $\gamma \notin \mathrm{Cn}(K * \Theta)$, and therefore $\{\gamma\} \not\subseteq \mathrm{Cn}(K * \Theta)$.

By hypothesis still, $\mathrm{Cn}(K \cup \Theta) = \mathcal{L}$.

So from Property A.2.1.1, replacing Γ_1 by $K * \Theta$, Γ_2 by $\{\gamma\}$, Γ_3 by K and $\overline{\Gamma_3}$ by Θ, there are $\overline{\{\gamma\}}$ and $\overline{\Theta}$ such that:

- $\mathrm{Cn}(\{\gamma\} \cup \overline{\{\gamma\}}) = \mathcal{L}$
- $\mathrm{Cn}((K * \Theta) \cup \overline{\{\gamma\}}) \neq \mathcal{L}$
- $\mathrm{Cn}(\overline{\Theta} \cup \Theta) = \mathcal{L}$
- $\overline{\Theta} \subseteq \mathrm{Cn}(K)$
- $\overline{\{\gamma\}} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})) \cup \Theta)$

The four conditions above are then verified for $\Omega = \mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})$, i.e. for $K' = \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})))$.

First, we can show that $K' \subseteq \mathrm{Cn}(K)$:

(1)	$\overline{\Theta} \subseteq \mathrm{Cn}(K)$	from the definition of $\overline{\Theta}$
(2)	$\mathrm{Cn}(\overline{\Theta}) \subseteq \mathrm{Cn}(\mathrm{Cn}(K))$	from (1), by monotonicity
(3)	$\mathrm{Cn}(\overline{\Theta}) \subseteq \mathrm{Cn}(K)$	from (2), by indempotence
(4)	$\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}) \subseteq \mathrm{Cn}(\overline{\Theta})$	
(5)	$\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}) \subseteq \mathrm{Cn}(K)$	from (3,4) and the transitivity of $\subseteq$

(6) $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq \mathrm{Cn}(K)$

(7) $((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \subseteq \mathrm{Cn}(K)$
from (5,6)

(8) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}))) \subseteq \mathrm{Cn}(K)$
from (7), by monotonicity and indempotence

(9) $K' \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})))$
from the definition of K'

(10) $K' \subseteq \mathrm{Cn}(K)$
from (8,9) and the transitivity of $\subseteq$

Then that $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K'$:

(1) $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq (\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}))$

(2) $(\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})))$
by (Tarskian) inclusion

(3) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}))) \subseteq K'$
from the definition of K'

(4) $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K'$ from (1,2,3) and the transitivity of $\subseteq$

Then that $\mathrm{Cn}(K' \cup \Theta) \neq \mathcal{L}$:

(1) $\Theta \subseteq K * \Theta$ by success

(2) $(K * \Theta) \cup \Theta = K * \Theta$ from (1)

(3) $\mathrm{Cn}((K * \Theta) \cup \Theta) = \mathrm{Cn}(K * \Theta)$ from (2)

(4) $\mathrm{Cn}(K * \Theta) = K * \Theta$ by closure

(5) $\mathrm{Cn}((K * \Theta) \cup \Theta) = K * \Theta$ from (3,4)

(6) $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K * \Theta$

(7) $(\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta \subseteq (K * \Theta) \cup \Theta$ from (6)

(8) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta) \subseteq \mathrm{Cn}((K * \Theta) \cup \Theta)$
from (7), by monotonicity

(9) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta) \subseteq K * \Theta$ from (5,8)

(10) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta) \cup \overline{\{\gamma\}} \subseteq (K * \Theta) \cup \overline{\{\gamma\}}$
from (9)

(11) $\mathrm{Cn}(\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta) \cup \overline{\{\gamma\}}) \subseteq \mathrm{Cn}((K * \Theta) \cup \overline{\{\gamma\}})$
from (10), by monotonicity

(12) $\mathrm{Cn}(\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta) \cup \overline{\{\gamma\}}) = \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta \cup \overline{\{\gamma\}})$
from theorem A.3.0.1

(13) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta \cup \overline{\{\gamma\}}) \subseteq \mathrm{Cn}((K * \Theta) \cup \overline{\{\gamma\}})$
from (11,12)

(14) $\mathrm{Cn}((K * \Theta) \cup \overline{\{\gamma\}}) \subset \mathcal{L}$ from the definition of $\overline{\{\gamma\}}$

(15) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \Theta \cup \overline{\{\gamma\}}) \subset \mathcal{L}$
from (13,14) and the transitivity of $\subseteq$

(16) $\mathrm{Cn}(\Theta) \cap \mathrm{Cn}(\overline{\{\gamma\}}) \subseteq \mathrm{Cn}(\overline{\{\gamma\}})$

(17) $(\mathrm{Cn}(\Theta) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta \subseteq \mathrm{Cn}(\overline{\{\gamma\}}) \cup \Theta$ from (16)

(18) $\mathrm{Cn}((\mathrm{Cn}(\Theta) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta) \subseteq \mathrm{Cn}(\mathrm{Cn}(\overline{\{\gamma\}}) \cup \Theta)$
from (17), by monotonicity

(19) $\mathrm{Cn}(\mathrm{Cn}(\overline{\{\gamma\}}) \cup \Theta) = \mathrm{Cn}(\overline{\{\gamma\}} \cup \Theta)$ from theorem A.3.0.1

(20) $\mathrm{Cn}((\mathrm{Cn}(\Theta) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta) \subseteq \mathrm{Cn}(\overline{\{\gamma\}} \cup \Theta)$
from (18,19)

(21) $K' \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})))$
from the definition of K'

(22) $K' \cup \Theta \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}))) \cup \Theta$

from (21)

(23) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}(\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}))) \cup \Theta)$

from (22), by monotonicity

(24) $\mathrm{Cn}(\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}}))) \cup \Theta) =$
$\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta)$

from theorem A.3.0.1

(25) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta)$

from (23,24)

(26) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup ((\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta))$

from (25)

(27) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup ((\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta)) =$
$\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}((\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta))$

from theorem A.3.0.1

(28) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}((\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta))$

from (26,27)

(29) $(\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}((\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta) \subseteq$
$(\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}(\overline{\{\gamma\}} \cup \Theta)$

from (20)

(30) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}((\mathrm{Cn}(\overline{\Theta}) \cap \mathrm{Cn}(\overline{\{\gamma\}})) \cup \Theta)) \subseteq$
$\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}(\overline{\{\gamma\}} \cup \Theta))$

from (29), by monotonicity

(31) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}(\overline{\{\gamma\}} \cup \Theta))$

from (28,30) and the transitivity of $\subseteq$

(32) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \mathrm{Cn}(\overline{\{\gamma\}} \cup \Theta)) = \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \overline{\{\gamma\}} \cup \Theta)$

from theorem A.3.0.1

(33) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup \overline{\{\gamma\}} \cup \Theta)$

from (31,32)

(34) $\mathrm{Cn}(K' \cup \Theta) \subset \mathcal{L}$ from (15,33) and the transitivity of $\subseteq$

(35) $\mathrm{Cn}(K' \cup \Theta) \neq \mathcal{L}$ from (34)

And finally that $\mathrm{Cn}(K' \cup \Theta \cup \{\gamma\}) = \mathcal{L}$:

(1) $\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta}) \subseteq (\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta}))$

(2) $(\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})) \subseteq \mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})))$

by (Tarskian) inclusion

(3) $\mathrm{Cn}((\mathrm{Cn}(K) \cap (K * \Theta)) \cup (\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta}))) \subseteq K'$

from the definition of K'

(4) $\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta}) \subseteq K'$ from (1,2,3) and the transitivity of $\subseteq$

(5) $(\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})) \cup \Theta \subseteq K' \cup \Theta$ from (4)

(6) $\mathrm{Cn}(\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})) \cup \Theta) \subseteq \mathrm{Cn}(K' \cup \Theta)$ from (5), by monotonicity

(7) $K' \cup \Theta \subseteq K' \cup \Theta \cup \{\gamma\}$

(8) $\mathrm{Cn}(K' \cup \Theta) \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ from (7), by monotonicity

(9) $\mathrm{Cn}(\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})) \cup \Theta) \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$

from (6,8) and the transitivity of $\subseteq$

(10) $\overline{\{\gamma\}} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\{\gamma\}}) \cap \mathrm{Cn}(\overline{\Theta})) \cup \Theta)$ from the definition of $\overline{\{\gamma\}}$

(11) $\overline{\{\gamma\}} \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ from (9,10) and the transitivity of $\subseteq$

(12) $\{\gamma\} \subseteq K' \cup \Theta \cup \{\gamma\}$

(13) $K' \cup \Theta \cup \{\gamma\} \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ by (Tarskian) inclusion

(14) $\{\gamma\} \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ from (12,13) and the transitivity of $\subseteq$

(15) $\mathrm{Cn}(\{\gamma\} \cup \overline{\{\gamma\}}) = \mathcal{L}$ from the definition of $\overline{\{\gamma\}}$

(16) $\overline{\{\gamma\}} \cup \{\gamma\} \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ from (11,14)

(17) $\mathrm{Cn}(\overline{\{\gamma\}} \cup \{\gamma\}) \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ from (16), by monotonicity and indempotence

(18) $\mathcal{L} \subseteq \mathrm{Cn}(K' \cup \Theta \cup \{\gamma\})$ from (15,17)

(19) $\mathrm{Cn}(K' \cup \Theta \cup \{\gamma\}) = \mathcal{L}$ from (18)

□

To sum up, if $< \mathcal{L}, \mathrm{Cn} >$ is Tarskian and if $*$ verifies closure, success and vacuity, then relevance is trivially verified if the cases where $K + \Theta \neq \mathcal{L}$ or $\mathrm{Cn}(\Theta) = \mathcal{L}$. In the only non trivial case where $K + \Theta = \mathcal{L}$ and $\mathrm{Cn}(\Theta) \neq \mathcal{L}$, if $< \mathcal{L}, \mathrm{Cn} >$ also satisfies Property A.2.1.1, then for each $\gamma \in \mathrm{Cn}(K) \setminus (K * \Theta)$, there is a K' such that:

- $K' \subseteq \mathrm{Cn}(K)$
- $\mathrm{Cn}(K) \cap (K * \Theta) \subseteq K'$
- $\mathrm{Cn}(K' \cup \Theta) \neq \mathcal{L}$
- $\mathrm{Cn}(K' \cup \Theta \cup \{\gamma\}) = \mathcal{L}$

Therefore in this case too, relevance is verified by $*$.

A.4 Proof of proposition A.2.2.1

Proposition. $\mathcal{ALCHO}$ under standard model-theoretic semantic for DLs and without unique name assumption verifies property A.2.1.1.

A.4.1 Lemmas

Lemma A.4.1.1. Let $< \mathcal{L}, \mathrm{Cn} >= \mathcal{ALCHO}$ and $\Gamma, \overline{\Gamma} \subseteq \mathcal{L}$, such that $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}) = \mathcal{L}$. Then there is a finite $\overline{\Gamma}' \subseteq \mathrm{Cn}(\overline{\Gamma})$ such that $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}') = \mathcal{L}$, and each $\gamma \in \overline{\Gamma}'$ is of

the form $C \sqsubseteq D$, with C and D DL concepts.

Proof. As a notational shortcut, if $\Omega \subseteq \mathcal{L}$, then $\mathrm{tb}(\Omega)$ and $\mathrm{rb}(\Omega)$ (for "TBox" and "RBox" respectively) will designate the formulas of Ω of the form $C \sqsubseteq D$ and $R \sqsubseteq S$ respectively, with C and D $\mathcal{ALCHO}$ concepts, and R and S DL atomic roles. As shown by the grammar of A-1, for any $\Omega \subseteq \mathcal{L}$, $\{\mathrm{tb}(\Omega), \mathrm{rb}(\Omega)\}$ is a partition of Ω.

Let $\Gamma, \overline{\Gamma} \subseteq \mathcal{L}$ and $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}) = \mathcal{L}$. We will first evacuate some limit cases.

If $\mathrm{Cn}(\Gamma) = \mathcal{L}$, set for instance $\overline{\Gamma}' = \{\top \sqsubseteq \top\}$. Then $\overline{\Gamma}' \subseteq \mathrm{Cn}\,\emptyset \subseteq \mathrm{Cn}(\overline{\Gamma})$, and $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}') = \mathrm{Cn}(\mathcal{L} \cup \overline{\Gamma}') = \mathcal{L}$.

If $\mathrm{Cn}(\overline{\Gamma}) = \mathcal{L}$, set for instance $\overline{\Gamma}' = \{\top \sqsubseteq \bot\}$. Then $\overline{\Gamma}' \subseteq \mathcal{L} \subseteq \mathrm{Cn}(\overline{\Gamma})$, and $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}') = \mathrm{Cn}(\Gamma \cup \mathcal{L}) = \mathcal{L}$.

Now for the non trivial case where $\mathrm{Cn}(\Gamma) \neq \mathcal{L}$ and $\mathrm{Cn}(\overline{\Gamma}) \neq \mathcal{L}$, we will first show that $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = \mathcal{L}$. Then only we will show that there is a finite $\overline{\Gamma}' \subseteq \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$ such that $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}') = \mathcal{L}$. If this is the case, because $\overline{\Gamma}' \subseteq \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$ and $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \subseteq \mathrm{Cn}(\overline{\Gamma})$, $\overline{\Gamma}' \subseteq \mathrm{Cn}(\overline{\Gamma})$ must hold, and so lemma A.4.1.1 holds as well.

In order to show that $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = \mathcal{L}$, let us first evacuate the limit case where $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma})) \subseteq \mathrm{Cn}(\emptyset)$, i.e. where each $\gamma \in \mathrm{Cn}(\overline{\Gamma})$ is either a tautology or of the form $C \sqsubseteq D$, with C and D $\mathcal{ALCHO}$ concepts. Then showing that $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) = \mathcal{L}$ is almost immediate:

(1)	$\mathrm{Cn}(\overline{\Gamma}) \subseteq \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$	because $\{\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})), \mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))\}$ is a partition of $\mathrm{Cn}(\overline{\Gamma})$
(2)	$\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma})) \subseteq \mathrm{Cn}(\emptyset)$	by hypothesis
(3)	$\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{rb}(\mathrm{Cn}(\overline{\Gamma})) \subseteq \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{Cn}(\emptyset)$	from (2)
(4)	$\mathrm{Cn}(\overline{\Gamma}) \subseteq \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{Cn}(\emptyset)$	from (1,3) and the transitivity of $\subseteq$
(5)	$\mathrm{Cn}(\overline{\Gamma}) \cup \Gamma \subseteq \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{Cn}(\emptyset) \cup \Gamma$	from (4)

(6) $\mathrm{Cn}(\mathrm{Cn}(\overline{\Gamma}) \cup \Gamma) \subseteq \mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{Cn}(\emptyset) \cup \Gamma)$
from (5), by monotonicity

(7) $\mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \mathrm{Cn}(\emptyset) \cup \Gamma) = \mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \emptyset \cup \Gamma) = \mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \Gamma)$
from theorem A.3.0.1

(8) $\mathrm{Cn}(\mathrm{Cn}(\overline{\Gamma}) \cup \Gamma) \subseteq \mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \Gamma)$ from (6,7)

(9) $\mathrm{Cn}(\overline{\Gamma} \cup \Gamma) = \mathrm{Cn}(\mathrm{Cn}(\overline{\Gamma}) \cup \Gamma)$ from theorem A.3.0.1

(10) $\mathrm{Cn}(\overline{\Gamma} \cup \Gamma) \subseteq \mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \Gamma)$ from (8,9)

(11) $\mathrm{Cn}(\overline{\Gamma} \cup \Gamma) = \mathcal{L}$ by hypothesis

(12) $\mathcal{L} \subseteq \mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \Gamma)$ from (10,11)

(13) $\mathrm{Cn}(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) \cup \Gamma) = \mathcal{L}$ from (12)

So the last remaining case is the one where $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma})) \not\subseteq \mathrm{Cn}(\emptyset)$. Let us assume by contradiction that $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \neq \mathcal{L}$. We will show that if this holds, then $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}) \neq \mathcal{L}$, contradicting the above hypothesis, and therefore $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = \mathcal{L}$ must hold.

$\mathcal{ALCHO}$ has the finite model property. So if $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \neq \mathcal{L}$, then there must be a finite model $\mathcal{I}_0 =< \Delta^{\mathcal{I}_0}, .^{\mathcal{I}_0} >$ of Γ such that $\mathcal{I}_0$ is also a model of $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$. Let $\mathrm{dom}(.^{\mathcal{I}_0})$ designate the domain of the interpretation function $.^{\mathcal{I}_0}$. $\mathcal{I}_0$ is finite means here that both $\Delta^{\mathcal{I}_0}$ and $\mathrm{dom}(.^{\mathcal{I}_0})$ are finite.

By hypothesis, $\mathrm{Cn}(\overline{\Gamma} \cup \Gamma) = \mathcal{L}$, so $\overline{\Gamma}$ and Γ do not share a model, and because $\mathcal{I}_0$ is a model of Γ, it cannot be a model of $\overline{\Gamma}$. So there must be a $\gamma \in \mathrm{Cn}(\overline{\Gamma})$ such that $\mathcal{I}_0 \not\models \gamma$. But because $\mathcal{I}_0$ is a model of $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$, for all $\gamma' \in \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$, $\mathcal{I}_0 \models \gamma'$ must hold. Therefore $\gamma \in \mathrm{Cn}(\overline{\Gamma}) \setminus \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})) = \mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$, i.e. γ must be of the form $R_1 \sqsubseteq S_1$, with R_1 and S_1 atomic DL roles.

So we have $\gamma = R_1 \sqsubseteq S_1$ and $\mathcal{I}_0 \not\models \gamma$, such that there must be $\delta_1, \delta_2 \in \Delta^{\mathcal{I}_0}$

verifying $(\delta_1, \delta_2) \in R_1^{\mathcal{I}_0} \setminus S_1^{\mathcal{I}_0}$. Now take the model $\mathcal{I}_1$ identical to $\mathcal{I}_0$, but with $S^{\mathcal{I}_1} = S^{\mathcal{I}_0} \cup \{(\delta_1, \delta_2)\}$. Then $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}_0}$, and if $\mathrm{dom}(.^{\mathcal{I}_0})$ and $\mathrm{dom}(.^{\mathcal{I}_1})$ are the respective domains of the interpretation functions $.^{\mathcal{I}_0}$ and $.^{\mathcal{I}_1}$, $\mathrm{dom}(.^{\mathcal{I}_1}) = \mathrm{dom}(.^{\mathcal{I}_0})$ also holds. Let $N_{Ind}(\mathrm{dom}(.^{\mathcal{I}_0}))$, $N_{Con}(\mathrm{dom}(.^{\mathcal{I}_0}))$ and $N_{Role}(\mathrm{dom}(.^{\mathcal{I}_0}))$ respectively designate the individuals, atomic DL concepts and atomic DL roles in $\mathrm{dom}(.^{\mathcal{I}_0}) = \mathrm{dom}(.^{\mathcal{I}_1})$. Then for each $e \in N_{Ind}(\mathrm{dom}(.^{\mathcal{I}_0}))$, $e^{\mathcal{I}_1} = \{e\}^{\mathcal{I}_1} = e^{\mathcal{I}_0} = \{e\}^{\mathcal{I}_0}$, for each $A \in N_{Con}(\mathrm{dom}(.^{\mathcal{I}_0}))$, $A^{\mathcal{I}_1} = A^{\mathcal{I}_0}$, and for each $R \in N_{Role}(\mathrm{dom}(.^{\mathcal{I}_0})) \setminus \{S\}$, $R^{\mathcal{I}_1} = R^{\mathcal{I}_0}$.

Let D be any (possibly complex) $\mathcal{ALCHO}$ concept. We will show by induction on the maximal number $\mathrm{maxNestQ}(D)$ of nested quantifiers in D that for each $\delta \in \Delta^{\mathcal{I}_0}$, $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$. Let $\delta \in \Delta^{\mathcal{I}_0} = \Delta^{\mathcal{I}_1}$. For the base case $\mathrm{maxNestQ}(D) = 0$, from the above observations, if $D = \{e_1, .., e_n\}$, with $\{e_1, .., e_n\} \subseteq N_{Ind}(\mathrm{dom}(.^{\mathcal{I}_0}))$, then $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$. Similarly, if $D \in N_{Con}(\mathrm{dom}(.^{\mathcal{I}_0}))$, then $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$ as well. Now let C_1, C_2 be two (possibly complex) DL concepts such that $\delta \in C_j^{\mathcal{I}_0}$ iff $\delta \in C_j^{\mathcal{I}_1}$. Then immediately from the semantic of the operators $\neg, \sqcap$ and $\sqcup$, $\delta \in (\neg C_1)^{\mathcal{I}_0}$ iff $\delta \in (\neg C_1)^{\mathcal{I}_1}$, $\delta \in (C_1 \sqcap C_2)^{\mathcal{I}_0}$ iff $\delta \in (C_1 \sqcap C_2)^{\mathcal{I}_1}$, and $\delta \in (C_1 \sqcup C_2)^{\mathcal{I}_0}$ iff $\delta \in (C_1 \sqcup C_2)^{\mathcal{I}_1}$. Because it is assumed that $\mathrm{maxNestQ}(D) = 0$, there is no quantifier in D. Therefore $\neg, \sqcap$ and $\sqcup$ are the only available operators to build a complex concept out of atomic concepts and sets of nominals, i.e. out of $N_{Con}(\mathrm{dom}(.^{\mathcal{I}_0})) \cup 2^{N_{Ind}(\mathrm{dom}(.^{\mathcal{I}_0}))}$. So by induction on the number of such operator in D, $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$ must hold.

For the inductive case now, let us start with the (sub)case where $D = \exists R.D'$ for some atomic DL role R and some $\mathcal{ALCHO}$ concept D'. If $\mathrm{maxNestQ}(D) = i$, then $\mathrm{maxNestQ}(D') = i - 1$ must hold. Let us start with the left direction, i.e. we need to show that if $\delta \in D^{\mathcal{I}_0}$, then $\delta \in D^{\mathcal{I}_1}$. If $\delta \in D^{\mathcal{I}_0}$, then there must be a $\delta' \in \Delta^{\mathcal{I}_0} = \Delta^{\mathcal{I}_1}$ such that $(\delta, \delta') \in R^{\mathcal{I}_0}$ and $\delta' \in (D')^{\mathcal{I}_0}$. From the construction of $\mathcal{I}_1$, if $R = S_1$, then $R^{\mathcal{I}_0} \subset R^{\mathcal{I}_1}$, and if $R \neq S_1$, then $R^{\mathcal{I}_0} = R^{\mathcal{I}_1}$. So in both cases, $R^{\mathcal{I}_0} \subseteq R^{\mathcal{I}_1}$ holds, and

therefore $(\delta, \delta') \in R^{\mathcal{I}_1}$. Then because $\delta' \in (D')^{\mathcal{I}_0}$, and because $\text{maxNestQ}(D') = i-1$, by IH, $\delta' \in (D')^{\mathcal{I}_1}$ must hold, and therefore $\delta \in (\exists R.D')^{\mathcal{I}_1} = D^{\mathcal{I}_1}$.

For the left direction now, let $\delta \in D^{\mathcal{I}_1}$. Then we need to show that $\delta \in D^{\mathcal{I}_0}$. Because $\delta \in D^{\mathcal{I}_1} = (\exists R.D')^{\mathcal{I}_1}$, there must be a $\delta' \in \Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}_0}$ such that $(\delta_1, \delta') \in R^{\mathcal{I}_1}$ and $\delta' \in (D')^{\mathcal{I}_1}$. From the construction of $\mathcal{I}_1$, if $R \neq S_1$, then $R^{\mathcal{I}_0} = R^{\mathcal{I}_1}$, so $R^{\mathcal{I}_0} \supseteq R^{\mathcal{I}_1}$ holds, and therefore $(\delta, \delta') \in R^{\mathcal{I}_0}$. Then similarly to the previous case, because by IH $\delta' \in (D')^{\mathcal{I}_0}$ also holds, $\delta \in (\exists R.D')^{\mathcal{I}_0} = D^{\mathcal{I}_0}$ is verified. If $R = S_1$ and either $\delta \neq \delta_1$ or there is a $\delta' \in (D')^{\mathcal{I}_1}$ such that $(\delta, \delta') \in S_1^{\mathcal{I}_1}$, and $\delta' \neq \delta_2$, then from the construction of $\mathcal{I}_1$, $(\delta, \delta') \in S_1^{\mathcal{I}_0}$, and again, because $\delta' \in (D')^{\mathcal{I}_0}$ by IH, $\delta \in (\exists S_1.D')^{\mathcal{I}_0} = D^{\mathcal{I}_0}$ is verified. So the last possible subcase is the one where $R = S_1$, $\delta = \delta_1$, $\delta_2 \in (D')^{\mathcal{I}_1}$, and there is not δ'' such that $(\delta, \delta'') \in S_1^{\mathcal{I}_1}$ and $\delta'' \in (D')^{\mathcal{I}_1}$. By IH, because $\delta_2 \in (D')^{\mathcal{I}_1}$, $\delta_2 \in (D')^{\mathcal{I}_0}$ must hold. And by hypothesis, $(\delta_1, \delta_2) \in R_1^{\mathcal{I}_0}$. So $\delta_1 \in (\exists R_1.D')^{\mathcal{I}_0}$. Now take the formula $\psi = \exists R_1.D' \sqsubseteq \exists S_1.D'$. Then $\psi \in \text{Cn}(\{R_1 \sqsubseteq S_1\})$. And because by hypothesis $R_1 \sqsubseteq S_1 \in \text{Cn}(\overline{\Gamma})$ holds, by monotonicity and indempotence, $\text{Cn}(\{R_1 \sqsubseteq S_1\}) \subseteq \text{Cn}(\overline{\Gamma})$ holds as well, such that $\psi \in \text{tb}(\text{Cn}(\overline{\Gamma}))$. Now by hypothesis still, $\mathcal{I}_0$ is a model of $\text{tb}(\text{Cn}(\overline{\Gamma}))$, and therefore $\mathcal{I}_0 \models \psi$ must hold. Then because $\delta_1 \in (\exists R_1.D')^{\mathcal{I}_0}$ and $\mathcal{I}_0 \models \exists R_1.D' \sqsubseteq \exists S_1.D'$, $\delta_1 \in (\exists S_1.D')_0^{\mathcal{I}} = D^{\mathcal{I}_0}$ is verified.

So by IH, if $\text{maxNestQ}(D) = i - 1$, for all $\delta \in \Delta^{\mathcal{I}_0} = \Delta^{\mathcal{I}_1}$, $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$ holds. And we have just shown if $D = \exists R.D'$, where $\text{maxNestQ}(D') = i - 1$, then $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$ holds as well. Now let C, C_1, C_2 be DL concepts such that $\delta \in C_j^{\mathcal{I}_0}$ iff $\delta \in C_j^{\mathcal{I}_1}$. Then immediately from the semantic of the operators $\neg, \sqcap$ and $\sqcup$, $\delta \in (\neg C_1)^{\mathcal{I}_0}$ iff $\delta \in (\neg C_1)^{\mathcal{I}_1}$, $\delta \in (C_1 \sqcap C_2)^{\mathcal{I}_0}$ iff $\delta \in (C_1 \sqcap C_2)^{\mathcal{I}_1}$, and $\delta \in (C_1 \sqcup C_2)^{\mathcal{I}_0}$ iff $\delta \in (C_1 \sqcup C_2)^{\mathcal{I}_1}$. If $D = \forall R.D'$, then D is equivalent to $\neg(\exists R.\neg D')$. So if $\text{maxNestQ}(D) = i$, from the grammar of figure A-1, D must be either of the form $\exists R.D'$, where $\text{maxNestQ}(D') = i - 1$, or equivalent to a concept of the form

$\neg C$, $C_1 \sqcap C_2$ or $C_1 \sqcup C_2$, such that $\text{maxNestQ}(C) = i$ holds, or at least one of $\text{maxNestQ}(C_1) = i$ and $\text{maxNestQ}(C_2) = i$ holds. So by induction on the number of operators of the form $\neg, \sqcap, \sqcup$ in D which are not part of a quantified subconcept of D, for any D such that $\text{maxNestQ}(D) = i$, for each $\delta \in \Delta^{\mathcal{I}_0} = \Delta^{\mathcal{I}_1}$, $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$ must hold.

We have just shown that for any (possibly complex) $\mathcal{ALCHO}$ concept D, for all $\delta \in \Delta^{\mathcal{I}_0}$, $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_1}$. Therefore for any formula of the form $C_1 \sqsubseteq C_2$, with C_1 and C_2 two $\mathcal{ALCHO}$ concepts, $\mathcal{I}_0 \models C_1 \sqsubseteq C_2$ iff $\mathcal{I}_1 \models C_1 \sqsubseteq C_2$. Now because $\mathcal{I}_0$ is a model of Γ, $\mathcal{I}_0$ is also model of $\text{Cn}(\Gamma)$, and because $\text{tb}(\text{Cn}(\Gamma)) \subseteq \text{tb}(\text{Cn}(\Gamma))$, $\mathcal{I}_0$ is a model of $\text{tb}(\text{Cn}(\Gamma))$. So for each $\gamma' \in \text{tb}(\text{Cn}(\Gamma))$, $\mathcal{I}_0 \models \gamma'$, and therefore $\mathcal{I}_1 \models \gamma'$ as well, such that $\mathcal{I}_1$ is a model of $\text{tb}(\text{Cn}(\Gamma))$ too. Similarly, because by hypothesis $\mathcal{I}_0$ is a model of $\text{tb}(\text{Cn}(\overline{\Gamma}))$, $\mathcal{I}_1$ must be a model of $\text{tb}(\text{Cn}(\overline{\Gamma}))$ too.

As a reminder, $\mathcal{I}_1$ is the model identical to $\mathcal{I}_0$, but with $S_1^{\mathcal{I}_1} = S_1^{\mathcal{I}_0} \cup \{(\delta_1, \delta_2)\}$ for some pair $(\delta_1, \delta_2) \in R_1^{\mathcal{I}_0} \setminus S_1^{\mathcal{I}_0}$. Now let us repeat the operation for each other pair in $R_1^{\mathcal{I}_0} \setminus S_1^{\mathcal{I}_0}$. Because $\mathcal{I}_0$ is finite, the set $\{(\delta_1^1, \delta_2^1), .., (\delta_1^m, \delta_2^m)\}$ of all such pairs is finite, so this yields a finite list of m interpretations $\mathcal{I}_0, .., \mathcal{I}_m$, such that for each $0 < i \leq m$, $\mathcal{I}_i$ is identical to $\mathcal{I}_{i-1}$, but with $S_1^{\mathcal{I}_i} = S_1^{\mathcal{I}_{i-1}} \cup \{(\delta_1^i, \delta_2^i)\}$. After termination, the interpretation $\mathcal{I}_m$ must be such that $R_1^{\mathcal{I}_m} = S_1^{\mathcal{I}_m}$, and therefore $\mathcal{I}_m \models R_1 \sqsubseteq S_1$ must hold.

Then this operation can be repeated for each other $R_j \sqsubseteq S_j \in \text{rb}(\text{Cn}(\overline{\Gamma}))$ such that $\mathcal{I}_0 \not\models R_j \sqsubseteq S_j$. Because $\mathcal{I}_0$ is finite, $\text{dom}(.^{\mathcal{I}}_0)$ is finite, and so there is a finite set of such formulas. Eventually, this yields $\mathcal{I}_n$ such that $\mathcal{I}_n$ is a model of $\text{rb}(\text{Cn}(\overline{\Gamma}))$. Although $\mathcal{I}_0$ is a model of $\text{rb}(\text{Cn}(\Gamma))$, it may be the case that $\mathcal{I}_n$ is not a model of $\text{rb}(\text{Cn}(\Gamma))$, because there is some $\gamma \in \text{rb}(\text{Cn}(\Gamma))$ such that $\mathcal{I}_n \not\models \gamma$. In this case, the whole operation can be repeated for each $\gamma \in \text{rb}(\text{Cn}(\Gamma))$ such that $\mathcal{I}_n \not\models \gamma$, yielding a model $\mathcal{I}_k$ of $\text{rb}(\text{Cn}(\Gamma))$, but not necessarily of $\text{rb}(\text{Cn}(\overline{\Gamma}))$. And once again

for each $\gamma \in \mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$ such that $\mathcal{I}_k \not\models \gamma$, etc, until the resulting interpretation is a model of both $\mathrm{rb}(\mathrm{Cn}(\Gamma))$ and $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$. The process must terminate, because each model $\mathcal{I}_i =< \Delta^{\mathcal{I}_i}, .^{\mathcal{I}_i} >$ produced during the operation is such that $\Delta^{\mathcal{I}_i} = \Delta^{\mathcal{I}_0}$, $\mathrm{dom}(.^{\mathcal{I}_i}) = \mathrm{dom}(.^{\mathcal{I}_0})$, and $S^{\mathcal{I}_i} = S^{\mathcal{I}_{i-1}} \cup \{(\delta_1, \delta_2)\}$ for some role $S \in \mathrm{dom}(.^{\mathcal{I}_0})$ and $\delta_1, \delta_2 \in \Delta^{\mathcal{I}_0}$, while $x^{\mathcal{I}_i} = x^{\mathcal{I}_{i-1}}$ for all $x \in \mathrm{dom}(.^{\mathcal{I}_0}) \setminus \{S\}$. So in the worst case, the process terminates with an interpretation $\mathcal{I}_l$ such that $S^{\mathcal{I}_l} = \Delta^{\mathcal{I}_0} \times \Delta^{\mathcal{I}_0}$ for each role $S \in \mathrm{dom}(.^{\mathcal{I}_0})$, and trivially, $\mathcal{I}_l$ verifies $R \sqsubseteq S$ for each possible pair (R, S) of roles in $\mathrm{dom}(.^{\mathcal{I}_0})$, So $\mathcal{I}_l$ must be a model of both $\mathrm{rb}(\mathrm{Cn}(\Gamma))$ and $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$. If the process terminates earlier, the resulting interpretation $\mathcal{I}_l$ must be a model of both $\mathrm{rb}(\mathrm{Cn}(\Gamma))$ and $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$ as well, precisely because this is the termination condition. Now by induction on l, for each $\delta \in \Delta^{\mathcal{I}_0} = \Delta^{\mathcal{I}_l}$, for any DL concept D, $\delta \in D^{\mathcal{I}_0}$ iff $\delta \in D^{\mathcal{I}_l}$, and because $\mathcal{I}_0$ is a model of both $\mathrm{tb}(\mathrm{Cn}(\Gamma))$ and $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$, $\mathcal{I}_l$ must be a model of both $\mathrm{tb}(\mathrm{Cn}(\Gamma))$ and $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$ as well. So $\mathcal{I}_l$ is a model of $\mathrm{tb}(\mathrm{Cn}(\Gamma))$, $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$, $\mathrm{rb}(\mathrm{Cn}(\Gamma))$ and $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma}))$, or in other words, $\mathcal{I}_l$ is a model of both Γ and $\overline{\Gamma}$, which contradict the hypothesis $\mathrm{Cn}(\Gamma \cup \overline{\Gamma}) = \mathcal{L}$.

So we have shown by contradiction that in the last remaining case where $\mathrm{rb}(\mathrm{Cn}(\overline{\Gamma})) \not\subseteq \mathrm{Cn}(\emptyset)$, if $\mathrm{Cn}(\Gamma \cup \mathrm{Cn}(\overline{\Gamma})) = \mathcal{L}$, then $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = \mathcal{L}$ must hold as well. Now we still need to show that there is a finite subset of $\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$ such that $\mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = \mathcal{L}$. $\mathcal{ALCHO}$ verifies compactness, or in other words, for each $\Gamma_1, \Gamma_2 \subseteq \mathcal{L}$, if $\Gamma_2 \subseteq \mathrm{Cn}(\Gamma_1)$, then there is a finite $s(\Gamma_1) \subseteq \Gamma_1$ such that $\Gamma_2 \subseteq \mathrm{Cn}(s(\Gamma_1))$. So because $\mathcal{L} \subseteq \mathrm{Cn}(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))$, there is a finite $s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \subseteq \Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))$ such that $\mathcal{L} \subseteq \mathrm{Cn}(s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))))$. Let $s(\Gamma) = \Gamma \cap s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))$, and let $s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}) \cap s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))))$. Then because $s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))$ is finite, $s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))$ is finite, and $\mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup \Gamma) = \mathcal{L}$ can be proven as follows:

(1) $s(\Gamma) \subseteq \Gamma$ from the definition of $s(\Gamma)$

(2) $\Gamma \subseteq \mathrm{Cn}(\Gamma)$ by (Tarskian) inclusion

(3) $(\Gamma) \subseteq \mathrm{Cn}(\Gamma)$ from (1,2) and the transitivity of $\subseteq$

(4) $s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup s(\Gamma) \subseteq s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup \mathrm{Cn}(\Gamma)$
from (3)

(5) $\mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup s(\Gamma)) \subseteq \mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup \mathrm{Cn}(\Gamma))$
from (4), by monotonicity

(6) $s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) = s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup s(\Gamma)$ from the definitions of $s(\Gamma)$ and $s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))$

(7) $\mathrm{Cn}(s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))) = \mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup s(\Gamma))$
from (6)

(8) $\mathcal{L} \subseteq \mathrm{Cn}(s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))))$ from the definition of $s(\Gamma \cup \mathrm{tb}(\mathrm{Cn}(\overline{\Gamma})))$

(9) $\mathcal{L} \subseteq \mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup s(\Gamma))$ from (7,8)

(10) $\mathcal{L} \subseteq \mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup \mathrm{Cn}(\Gamma))$ from (5,9) and the transitivity of $\subseteq$

(11) $\mathrm{Cn}(s(\mathrm{tb}(\mathrm{Cn}(\overline{\Gamma}))) \cup \mathrm{Cn}(\Gamma)) = \mathcal{L}$ from (10)

□

A.4.2 Main proposition

Property A.2.1.1 is reproduced here for readability:

Property. Let $\Gamma_1, \Gamma_2, \Gamma_3, \overline{\Gamma_3} \subseteq \mathcal{L}$, with $\Gamma_2 \not\subseteq \mathrm{Cn}(\Gamma_1)$ and $\mathrm{Cn}(\Gamma_3 \cup \overline{\Gamma_3}) = \mathcal{L}$. Then there are $\overline{\Gamma_2}, \Gamma_4 \subseteq \mathcal{L}$ such that:

a: $\mathrm{Cn}(\Gamma_2 \cup \overline{\Gamma_2}) = \mathcal{L}$

b: $\mathrm{Cn}(\Gamma_1 \cup \overline{\Gamma_2}) \neq \mathcal{L}$

c: $\mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_3}) = \mathcal{L}$

d: $\Gamma_4 \subseteq \ \mathrm{Cn}(\Gamma_3)$

e: $\overline{\Gamma_2} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$

And this is the proposition to be proven:

Proposition. $\mathcal{ALCHO}$ under standard model-theoretic semantic for DLs and without unique name assumption verifies property A.2.1.1.

Proof. Let $< \mathcal{L}, \mathrm{Cn} > = \mathcal{ALCHO}$. From the grammar of figure A-1, if $\gamma \in \mathcal{L}$, then either $\gamma = C \sqsubseteq D$, with C and D two (possibly complex) DL concepts, or $\gamma = R \sqsubseteq S$, with R and S two DL roles.

Take any $\Gamma_1, \Gamma_2, \Gamma_3, \overline{\Gamma_3} \subseteq \mathcal{L}$ such that $\Gamma_2 \not\subseteq \mathrm{Cn}(\Gamma_1)$ and $\mathrm{Cn}(\overline{\Gamma_3} \cup \Gamma_3) = \mathcal{L}$. Will first focus on $\Gamma_1, \Gamma_2, \overline{\Gamma_2}$ and condition **a** and **b** of Property A.2.1.1.

Because $\Gamma_2 \not\subseteq \mathrm{Cn}(\Gamma_1)$, there is a $\gamma_2 \in \Gamma_2 \setminus \mathrm{Cn}(\Gamma_1)$, so $\mathrm{Cn}(\Gamma_1) \neq \mathcal{L}$, and so Γ_1 is consistent. Take a model $\mathcal{I} =< \Delta^{\mathcal{I}}, .^{\mathcal{I}} >$ of Γ_1. Then $\mathcal{I} \not\models \gamma_2$ must hold.

Let us assume that $\gamma_2 = C \sqsubseteq D$, with C and D DL concepts, then there is a $\delta \in C^{\mathcal{I}} \setminus D^{\mathcal{I}}$. If $\mathrm{dom}(.^{\mathcal{I}})$ is the domain of the interpretation function $.^{\mathcal{I}}$, take any DL individual $e \notin \mathrm{dom}(.^{\mathcal{I}})$, i.e. such that $.^{\mathcal{I}}$ is not defined for e, and let $\overline{\gamma_2} = \{e\} \sqsubseteq C \sqcap \neg D$. Then build the interpretation $\mathcal{I}' =< \Delta^{\mathcal{I}}, .^{\mathcal{I}'} >$, with $.^{\mathcal{I}'}$ defined by $\mathrm{dom}(.^{\mathcal{I}'}) = \mathrm{dom}(.^{\mathcal{I}}) \cup \{e\}$, $x^{\mathcal{I}'} = x^{\mathcal{I}}$ for all $x \in \mathrm{dom}(.^{\mathcal{I}})$, and $e^{\mathcal{I}'} = \delta$. Then $\mathcal{I}'$ is a model of Γ_1, and $\mathcal{I}' \models \overline{\gamma_2}$ also holds.

Now let $\overline{\Gamma_2} = \{\overline{\gamma_2}\}$. Because $\mathcal{I}'$ is a model of Γ_1 and $\mathcal{I}' \models \overline{\gamma_2}$, $\Gamma_1 \cup \overline{\Gamma_2}$ is satisfiable, i.e. $\mathrm{Cn}(\Gamma_1 \cup \overline{\Gamma_2}) \neq \mathcal{L}$. But $\{\gamma_2, \overline{\gamma_2}\} = \{C \sqsubseteq D, \{e\} \sqsubseteq C \sqcap \neg D\}$ is inconsistent, i.e. $\mathrm{Cn}(\{\gamma_2, \overline{\gamma_2}\}) = \mathrm{Cn}(\{\gamma_2\} \cup \overline{\Gamma_2}) = \mathcal{L}$.

Now let us assume instead that $\gamma_2 = R \sqsubseteq S$, with R and S DL atomic roles. Then similarly, there are $\delta_1, \delta_2 \in \Delta^{\mathcal{I}}$ such that $(\delta_1, \delta_2) \in R^{\mathcal{I}} \setminus S^{\mathcal{I}}$. Take any two DL individual $e_1, e_2 \notin \mathrm{dom}(.^{\mathcal{I}})$, and let $\overline{\gamma_2} = \{e_1\} \sqsubseteq (\exists R.\{e_2\}) \sqcap (\forall S.\neg\{e_2\})$. Then build

the interpretation $\mathcal{I}' =< \Delta^{\mathcal{I}}, .^{\mathcal{I}'} >$, with $.^{\mathcal{I}'}$ defined by $\text{dom}(.^{\mathcal{I}'}) = \text{dom}(.^{\mathcal{I}}) \cup \{e_1, e_2\}$, $x^{\mathcal{I}'} = x^{\mathcal{I}}$ for all $x \in \text{dom}(.^{\mathcal{I}})$, $e_1^{\mathcal{I}'} = \delta_1$ and $e_2^{\mathcal{I}'} = \delta_2$. Then $\mathcal{I}'$ is a model of Γ_1, and $\mathcal{I}' \models \overline{\gamma_2}$ also holds.

Now let $\overline{\Gamma_2} = \{\overline{\gamma_2}\}$, Because $\mathcal{I}'$ is a model of Γ_1 and $\mathcal{I}' \models \overline{\gamma_2}$, $\Gamma_1 \cup \overline{\Gamma_2}$ is satisfiable, i.e. $\text{Cn}(\Gamma_1 \cup \overline{\Gamma_2}) \neq \mathcal{L}$. But $\{\gamma_2, \overline{\gamma_2}\} = \{R \sqsubseteq S, \{e_1\} \sqsubseteq (\exists R.\{e_2\}) \sqcap (\forall R.\neg\{e_2\})\}$ is inconsistent, i.e. $\text{Cn}(\{\gamma_2, \overline{\gamma_2}\}) = \text{Cn}(\{\gamma_2\} \cup \overline{\Gamma_2}) = \mathcal{L}$.

So in both cases ($\gamma_2 = C \sqsubseteq D$ or $\gamma_2 = R \sqsubseteq S$), we have $\text{Cn}(\Gamma_1 \cup \overline{\Gamma_2}) \neq \mathcal{L}$, such that condition **b** of Property A.2.1.1 is verified by $\overline{\Gamma_2}$. And in both cases too, $\text{Cn}(\{\gamma_2\} \cup \overline{\Gamma_2}) = \mathcal{L}$ holds, such that condition **a** can be shown to hold as well, as follows:

(1)	$\gamma_2 \in \Gamma_2$	from the definition of γ_2
(2)	$\{\gamma_2\} \subseteq \Gamma_2$	from (1)
(3)	$\{\gamma_2\} \cup \overline{\Gamma_2} \subseteq \Gamma_2 \cup \overline{\Gamma_2}$	from (2)
(4)	$\text{Cn}(\{\gamma_2\} \cup \overline{\Gamma_2}) \subseteq \text{Cn}(\Gamma_2 \cup \overline{\Gamma_2})$	from (3), by monotonicity
(5)	$\text{Cn}(\{\gamma_2\} \cup \overline{\Gamma_2}) = \mathcal{L}$	from the above observation
(6)	$\mathcal{L} \subseteq \text{Cn}(\Gamma_2 \cup \overline{\Gamma_2})$	from (4,5)
(7)	$\text{Cn}(\Gamma_2 \cup \overline{\Gamma_2}) = \mathcal{L}$	from (6)

So we have shown that there is a $\overline{\Gamma_2}$ which verifies conditions **a** and **b** of Property A.2.1.1, and such that $\overline{\Gamma_2}$ is composed of only one formula $\overline{\gamma_2}$, of the form $\{e\} \sqsubseteq C \sqcap \neg D$, or $\{e_1\} \sqsubseteq (\exists R.\{e_2\}) \sqcap (\forall R.\neg\{e_2\})$.

Now let us focus on conditions **c** and **d**.

Let us first evacuate the limit case where $\text{Cn}(\Gamma_3) = \mathcal{L}$. Take $\Gamma_4 = \{\top \sqsubseteq \bot\}$. Then $\Gamma_4 \subseteq \mathcal{L} = \text{Cn}(\Gamma_3)$, so condition **d** is trivially verified. Similarly, $\text{Cn}(\Gamma_4 \cup \overline{\Gamma_3}) = \text{Cn}(\{\top \sqsubseteq \bot\} \cup \mathcal{L}) = \mathcal{L}$, such that condition **c** is verified as well. So we only need to

show that condition **e** holds, i.e. that $\overline{\Gamma_2} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$, which can be done as follows:

(1)	$\Gamma_4 = \{\top \sqsubseteq \bot\}$	by hypothesis
(2)	$\mathrm{Cn}(\Gamma_4) = \mathcal{L}$	from (1)
(3)	$\mathrm{Cn}(\overline{\Gamma_2}) \subseteq \mathcal{L}$	
(4)	$\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4) = \mathrm{Cn}(\overline{\Gamma_2})$	from (2,3)
(5)	$\overline{\Gamma_2} \subseteq \mathrm{Cn}(\overline{\Gamma_2})$	by (Tarskian) inclusion
(6)	$\overline{\Gamma_2} \subseteq \mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)$	from (4,5)
(7)	$\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4) \subseteq (\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$	
(8)	$(\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3}) \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3}))$	by (Tarskian) inclusion
(9)	$\overline{\Gamma_2} \subseteq (\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$	from (6,7,8) and the transitivity of $\subseteq$

The last possible case is the one where $\mathrm{Cn}(\Gamma_3) \neq \mathcal{L}$. By hypothesis, $\mathrm{Cn}(\overline{\Gamma_3} \cup \Gamma_3) = \mathcal{L}$, and so from lemma A.4.1.1, there is a finite $\Omega \subseteq \mathrm{Cn}(\Gamma_3)$ such that $\mathrm{Cn}(\Omega \cup \overline{\Gamma_3}) = \mathcal{L}$, and each $\omega \in \Omega$ is of the form $C \sqsubseteq D$, with C and D $\mathcal{ALCHO}$ concepts. Let $\Gamma_4 = \Omega$. Then Γ_4 verifies conditions **c** and **d**, namely $\mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_3}) = \mathcal{L}$ and $\Gamma_4 \subseteq \mathrm{Cn}(\Gamma_3)$. So we only need to show that condition **e** holds, i.e. that $\overline{\Gamma_2} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$.

From the the above propositions, either $\overline{\Gamma_2} = \{\{e_1\} \sqsubseteq C_1 \sqcap \neg C_2\}$, or $\overline{\Gamma_2} = \{\{e_1\} \sqsubseteq (\exists R.\{e_2\}) \sqcap (\forall R.\neg\{e_2\})\}$, so in both cases, $\overline{\Gamma_2}$ is composed of a single formula $\overline{\gamma_2}$ of the form $\{e_{\overline{\gamma_2}}\} \sqsubseteq D_{\overline{\gamma_2}}$, with $e_{\overline{\gamma_2}}$ an individual an $D_{\overline{\gamma_2}}$ a (complex) DL concept. From the above propositions still, $\Gamma_4 = \{C_1 \sqsubseteq D_1, .., C_n \sqsubseteq D_n\}$, with C_i, D_i DL concepts. Let $\gamma_4 = \top \sqsubseteq (\neg C_1 \sqcup D_1) \sqcap .. \sqcap (\neg C_n \sqcup D_n)$. Then $\mathrm{Cn}(\{\gamma_4\}) = \mathrm{Cn}(\Gamma_4)$. As a notational shortcut, let $C_{\gamma_4} = (\neg C_1 \sqcup D_1) \sqcap .. \sqcap (\neg C_n \sqcup D_n)$, such that $\gamma_4 = \top \sqsubseteq C_{\gamma_4}$. Now consider the formula $\gamma' = \top \sqsubseteq \neg\{e_{\overline{\Gamma_2}}\} \sqcup D_{\overline{\Gamma_2}} \sqcup C_{\gamma_4}$. Then $\gamma' \in \mathrm{Cn}(\{\overline{\gamma_2}\}) = \mathrm{Cn}(\overline{\Gamma_2})$,

and $\gamma' \in \mathrm{Cn}(\{\gamma_4\}) = \mathrm{Cn}(\Gamma_4))$, such that $\gamma' \in \mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)$.

Then because $\mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_3}) = \mathcal{L}$, from lemma A.4.1.1, there is a finite $\overline{\Gamma_4} \subseteq \mathrm{Cn}(\overline{\Gamma_3})$ such that $\mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_4}) = \mathcal{L}$, and each $\gamma \in \overline{\Gamma_4}$ is of the form $C' \sqsubseteq D'$, with C' and D' DL concepts. If $\overline{\Gamma_4} = \{C'_1 \sqsubseteq D'_1, .., C'_m \sqsubseteq D'_m\}$, with C'_i, D'_i DL concepts, let $\overline{\gamma_4} = \top \sqsubseteq (\neg C'_1 \sqcup D'_1) \sqcap .. \sqcap (\neg C'_m \sqcup D'_m)$. Then $\mathrm{Cn}(\{\overline{\gamma_4}\}) = \mathrm{Cn}(\overline{\Gamma_4})$. As a notational shortcut, let $C_{\overline{\gamma_4}} = (\neg C'_1 \sqcup D'_1) \sqcap .. \sqcap (\neg C'_m \sqcup D'_m)$, such that $\overline{\gamma_4} = \top \sqsubseteq C_{\overline{\gamma_4}}$. Now we can show that $\overline{\Gamma_2} \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$, as follows:

(1) $\mathrm{Cn}(\{\gamma_4\}) = \mathrm{Cn}(\Gamma_4)$

(2) $\mathrm{Cn}(\{\overline{\gamma_4}\}) = \mathrm{Cn}(\overline{\Gamma_4})$

(3) $\mathrm{Cn}(\{\gamma_4\}) \cup \mathrm{Cn}(\{\overline{\gamma_4}\}) = \mathrm{Cn}(\Gamma_4) \cup \mathrm{Cn}(\overline{\Gamma_4})$ from (1,2)

(4) $\mathrm{Cn}(\mathrm{Cn}(\{\gamma_4\}) \cup \mathrm{Cn}(\{\overline{\gamma_4}\})) = \mathrm{Cn}(\mathrm{Cn}(\Gamma_4) \cup \mathrm{Cn}(\overline{\Gamma_4}))$
from (3), by monotonicity

(5) $\mathrm{Cn}(\mathrm{Cn}(\{\gamma_4\}) \cup \mathrm{Cn}(\overline{\{\gamma_4\}})) = \mathrm{Cn}(\{\gamma_4\} \cup \{\overline{\gamma_4}\})$
from theorem A.3.0.1 (twice)

(6) $\mathrm{Cn}(\mathrm{Cn}(\Gamma_4) \cup \mathrm{Cn}(\overline{\Gamma_4})) = \mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_4})$ from theorem A.3.0.1 (twice)

(7) $\mathrm{Cn}(\{\gamma_4\} \cup \{\overline{\gamma_4}\}) = \mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_4})$ from (4,5,6)

(8) $\mathrm{Cn}(\Gamma_4 \cup \overline{\Gamma_4}) = \mathcal{L}$ from the definition of $\overline{\Gamma_4}$

(9) $\mathrm{Cn}(\{\gamma_4, \overline{\gamma_4}\}) = \mathcal{L}$ from (7,8)

(10) $\gamma_4 = \top \sqsubseteq C_{\gamma_4}$ from the definition of γ_4

(11) $\overline{\gamma_4} = \top \sqsubseteq C_{\overline{\gamma_4}}$ from the definition of $\overline{\gamma_4}$

(12) $\mathrm{Cn}(\{\top \sqsubseteq C_{\gamma_4}, \top \sqsubseteq C_{\overline{\gamma_4}}\}) = \mathcal{L}$ from (9,10,11)

(13) $\mathrm{Cn}(\{\top \sqsubseteq C_{\gamma_4} \sqcap C_{\overline{\gamma_4}}\}) = \mathcal{L}$ from (12)

(14) $C_{\gamma_4} \sqcap C_{\overline{\gamma_4}} \sqsubseteq \bot \in \mathrm{Cn}(\emptyset)$ from (13)

(15) $\gamma' = \top \sqsubseteq \neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}} \sqcup C_{\gamma_4}$ from the definition of γ'

(16) $\gamma' \in \mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)$ from the above observations

(17) $\overline{\gamma_4} \in \mathrm{Cn}(\{\overline{\gamma_4}\})$ by (Tarskian) inclusion

(18) $\overline{\Gamma_4} \subseteq \mathrm{Cn}(\overline{\Gamma_3})$ from the definition of $\overline{\Gamma_4}$

(19) $\mathrm{Cn}(\overline{\Gamma_4}) \subseteq \mathrm{Cn}(\overline{\Gamma_3})$ from (18), by monotonicity and indempotence

(20) $\overline{\gamma_4} \in \mathrm{Cn}(\overline{\Gamma_3})$ from (2,17,19)

(21) $\{\gamma', \overline{\gamma_4}\} \subseteq (\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \mathrm{Cn}(\overline{\Gamma_3})$ from (16,20)

(22) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \mathrm{Cn}(\overline{\Gamma_3}))$

from (21), by monotonicity

(23) $\mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \mathrm{Cn}(\overline{\Gamma_3})) = \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$

from theorem A.3.0.1

(24) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) \subseteq \mathrm{Cn}((\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \overline{\Gamma_3})$ from (22,23)

(25) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) = \mathrm{Cn}(\{\top \sqsubseteq \neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}} \sqcup C_{\gamma_4}, \top \sqsubseteq C_{\overline{\gamma_4}}\}$

from (11,15)

(26) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) = \mathrm{Cn}(\{\top \sqsubseteq (\neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}} \sqcup C_{\gamma_4}) \sqcap C_{\overline{\gamma_4}}\})$

from (25)

(27) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) = \mathrm{Cn}(\{\top \sqsubseteq (\neg\{e_{\overline{\gamma_2}}\} \sqcap C_{\overline{\gamma_4}}) \sqcup (D_{\overline{\gamma_2}} \sqcap C_{\overline{\gamma_4}}) \sqcup (C_{\gamma_4} \sqcap C_{\overline{\gamma_4}})\})$

from (26) and the distributivity of $\sqcap$ and $\sqcup$

(28) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) = \mathrm{Cn}(\{\top \sqsubseteq (\neg\{e_{\overline{\gamma_2}}\} \sqcap C_{\overline{\gamma_4}}) \sqcup (D_{\overline{\gamma_2}} \sqcap C_{\overline{\gamma_4}}) \sqcup \bot)\})$

from (14,24)

(29) $\mathrm{Cn}(\{\gamma', \overline{\gamma_4}\}) = \mathrm{Cn}(\{\top \sqsubseteq (\neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}}) \sqcap C_{\overline{\gamma_4}}\})$

from (28) and the distributivity of $\sqcap$ and $\sqcup$

(30) $\top \sqsubseteq \neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}} \in \mathrm{Cn}(\{\top \sqsubseteq (\neg\{e\} \sqcup D_{\overline{\gamma_2}}) \sqcap C_{\overline{\gamma_4}}\})$

(31) $\mathrm{Cn}(\{\top \sqsubseteq \neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}}\}) \subseteq \mathrm{Cn}(\{\top \sqsubseteq (\neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}}) \sqcap C_{\overline{\gamma_4}}\})$

from (30), by monotonicity and indempotence

(32) $\mathrm{Cn}(\{\top \sqsubseteq \neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}}) \subseteq \mathrm{Cn}(\{\gamma', \overline{\gamma_4}\})$ from (29,31)

(33) $\overline{\Gamma_2} = \{\top \sqsubseteq \neg\{e_{\overline{\gamma_2}}\} \sqcup D_{\overline{\gamma_2}}\}$ from the definition of $\overline{\Gamma_2}$

(34) $\mathrm{Cn}(\overline{\Gamma_2}) \subseteq \mathrm{Cn}(\{\gamma', \overline{\gamma_4}\})$ from (32,33)

(35) $\mathrm{Cn}(\overline{\Gamma_2}) \subseteq \mathrm{Cn}(\mathrm{Cn}(\overline{\Gamma_2}) \cap \mathrm{Cn}(\Gamma_4)) \cup \mathrm{Cn}(\overline{\Gamma_3}))$ from (22,34) and the transitivity of $\subseteq$

□

Appendix B

Foundational ontology, attachments and erroneous axioms

This appendix provides a more concrete illustration of the manual extension strategy described in Chapter 7, as well as examples of axioms which were considered as actually erroneous for the evaluations which are based on real datasets, and not automatically degraded ones.

Section B.1 reproduces the exhaustive taxonomy of the foundational ontology TMEO, introduced in Chapter 7 Section 7.4.3, which can be viewed as an illustration of a minimal foundational ontology for the manual extension strategy described in Chapter 7, whereas Section B.1.3 provides examples of manual attachments of elements of the signature of DBpedia to TMEO. Finally, Section B.2 provides a list of DBpedia axioms which were considered as actually erroneous by an ontology expert.

B.1 TMEO

TMEO is the ontology backing the categorization tool TMEO used in the Senso Comune project, in order to develop a lexical-ontological resource [JVZ+14]. An OWL version of TMEO (with comments) is available online.[1] All subsumption and disjointness axioms are reproduced in Section **??** and Section **??** respectively.

B.1.1 Subsumption axioms

Figures B-1, B-3 and B-2 reproduce all subsumption axioms in TMEO. Figure B-1 represents the more abstract categories, whereas Figure B-3 and Figure B-2 reproduce the categories transitively subsumed by `NonTangibleEntity` and `TangibleEntity` respectively.

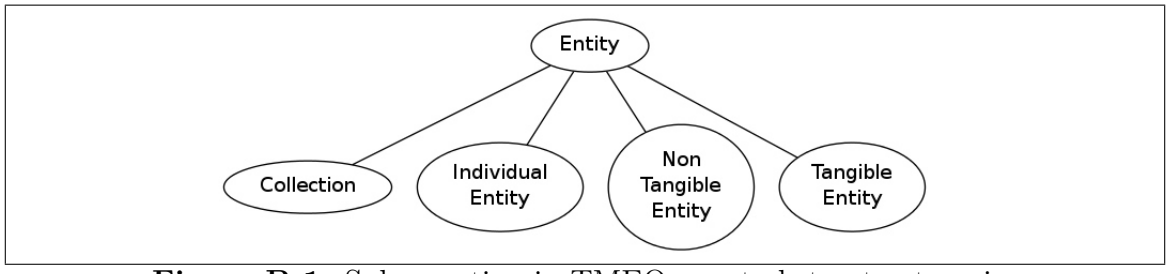

Figure B-1: Subsumption in TMEO: most abstract categories

[1] http://juliencorman.github.io

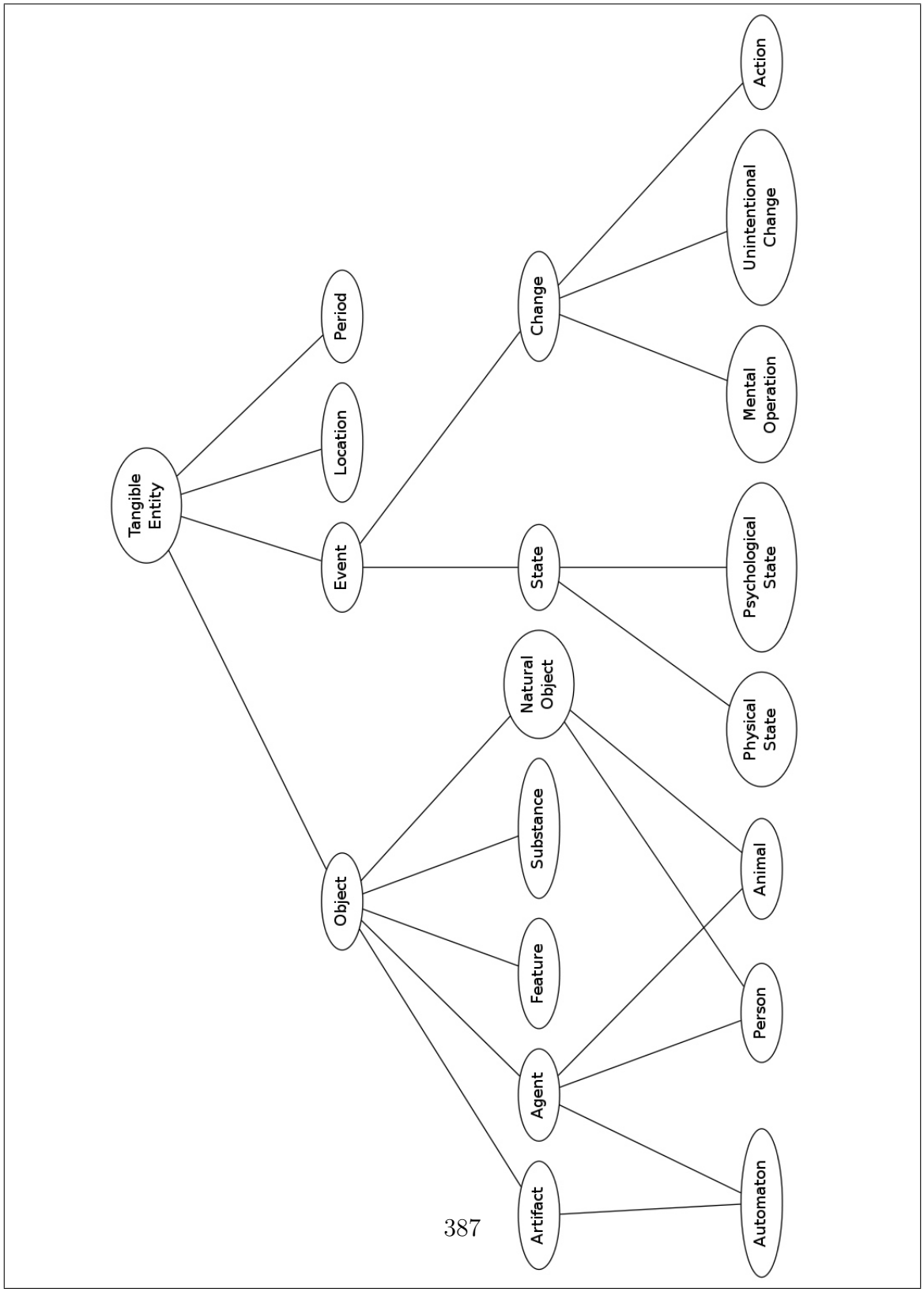

Figure B-2: Subsumption in TMEO: `TangibleEntity` and subsumed categories

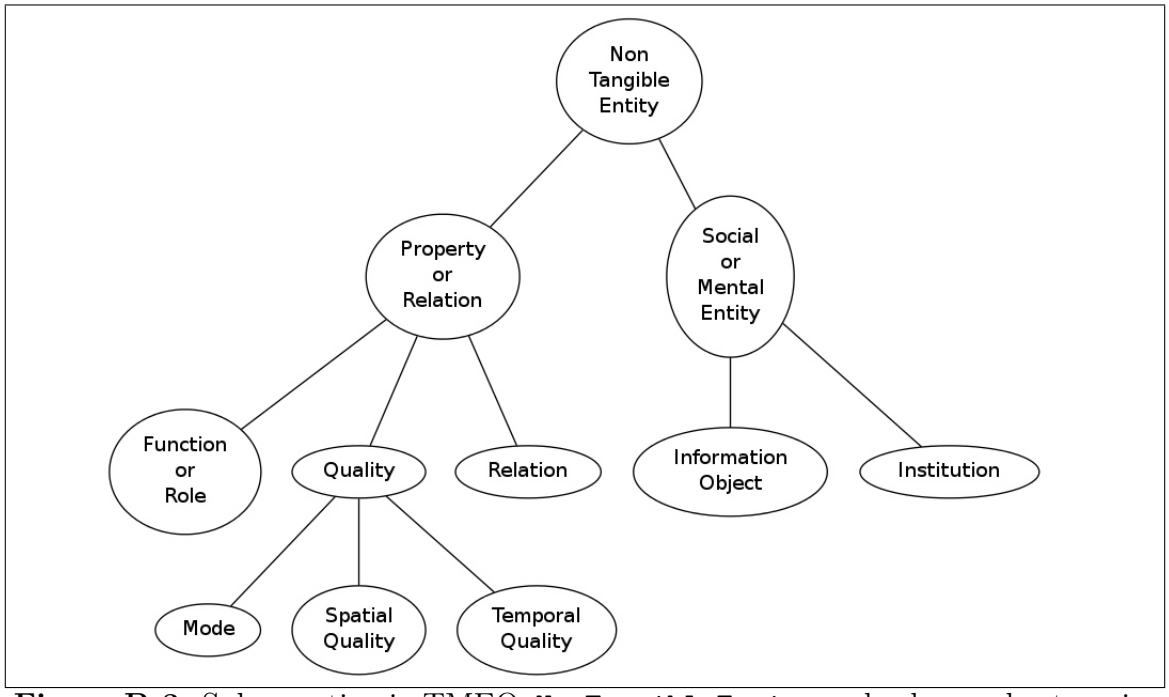

Figure B-3: Subsumption in TMEO: `NonTangibleEntity` and subsumed categories

B.1.2 Disjointess axioms

Here is the exhaustive list of disjointness axioms in TMEO:

$\texttt{Action} \sqsubseteq \neg\texttt{UnintentionalChange}$

$\texttt{Agent} \sqsubseteq \neg\texttt{Feature}$

$\texttt{Agent} \sqsubseteq \neg\texttt{Substance}$

$\texttt{Artifact} \sqsubseteq \neg\texttt{NaturalObject}$

$\texttt{Change} \sqsubseteq \neg\texttt{State}$

$\texttt{Collection} \sqsubseteq \neg\texttt{IndividualEntity}$

$\texttt{Event} \sqsubseteq \neg\texttt{Location}$

$\texttt{Event} \sqsubseteq \neg\texttt{Object}$

$\texttt{Event} \sqsubseteq \neg\texttt{Period}$

Feature ⊑ ¬Substance

FunctionOrRole ⊑ ¬Quality

FunctionOrRole ⊑ ¬Relation

IndividualEntity ⊑ ¬Mode

InformationObject ⊑ ¬Institution

InformationObject ⊑ ¬OtherMentalEntity

Institution ⊑ ¬OtherSocialEntity

Location ⊑ ¬Object

Location ⊑ ¬Period

Mode ⊑ ¬SpatialQuality

NonTangibleEntity ⊑ ¬TangibleEntity

Object ⊑ ¬Period

OtherNonTangible ⊑ ¬PropertyOrRelation

OtherNonTangible ⊑ ¬SocialOrMentalEntity

OtherState ⊑ ¬PhysicalState

OtherState ⊑ ¬PsychologicalState

PhysicalState ⊑ ¬PsychologicalState

PropertyOrRelation ⊑ ¬SocialOrMentalEntity

Quality ⊑ ¬Relation

SpatialQuality ⊑ ¬TemporalQuality

B.1.3 Attachment: examples

The following are some examples of manual attachments to TMEO of DBpedia individuals, atomic concepts, datatypeProperties (treated as atomic concepts) or objectProperties, as described in Chapter 7 Section 7.3.1. Elements of the signature

of DBpedia are prefixed with "DBP:", and elements of the signature of TMEO are prefixed with "TMEO:", and colored in green.

Each line or pair of lines if place is missing) represents an attachment axiom, where the element to be attached is on the left, and the attachment axiom on the right.

B.1.3.1 Individuals

DBP:Mohamed Al-Fayed TMEO:Person(*DBP:Mohamed Al-Fayed*)

DBP:Birds of South Asia.The Ripley Guide

TMEO:InformationObject(*DBP:Birds of South Asia.The Ripley Guide*)

DBP:Smithsonian Folkways TMEO:Institution(*DBP:Smithsonian Folkways*)

DBP:Andrew Carnegie Mansion

TMEO:TMEO:TangibleEntity(*DBP:Andrew Carnegie Mansion*)

B.1.3.2 Atomic concepts

DBP:Building $\sqsubseteq$ TMEO:TangibleEntity

DBP:PeriodicalLiterature

DBP:PeriodicalLiterature $\sqsubseteq$ TMEO:InformationObject

DBP:Person $\sqsubseteq$ TMEO:Person

DBP:Organisation $\sqsubseteq$ TMEO:Institution

DBP:SoccerClubSeason $\sqsubseteq$ TMEO:Change

B.1.3.3 DatatypeProperties

DBP:deathDate $\text{HAS}_{deathDate} \sqsubseteq$ TMEO:Person

DBP:yearOfConstruction $\text{HAS}_{yearOfConstruction} \sqsubseteq$ TMEO:TangibleEntity

DBP:added	$\texttt{HAS}_{added} \sqsubseteq$ TMEO:TangibleEntity
DBP:managerTitle	$\texttt{HAS}_{managerTitle} \sqsubseteq$ TMEO:Institution

B.1.3.4 ObjectProperties

DBP:occupation	$\top \sqsubseteq \forall$DBP:occupation.TMEO:RoleOrFunction
DBP:occupation	$\exists$DBP:occupation.$\top \sqsubseteq$ TMEO:Person
DBP:spouse	$\top \sqsubseteq \forall$DBP:spouse.TMEO:Person
DBP:spouse	$\exists$DBP:spouse.$\top \sqsubseteq$ TMEO:Person
DBP:nationality	$\top \sqsubseteq \forall$DBP:nationality.TMEO:SocialOrMentalEntity
DBP:nationality	$\exists$DBP:nationality.$\top \sqsubseteq$ TMEO:Person
DBP:governingBody	$\top \sqsubseteq \forall$DBP:governingBody.TMEO:Institution
DBP:governingBody	$\exists$DBP:governingBody.$\top \sqsubseteq$ TMEO:Location

B.2 Erroneous axioms: examples

The following is the list of axioms manually identified as erroneous in the KBs $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$ by the formal ontology expert. $K_{1.1}^{DBP}$ and $K_{1.2}^{DBP}$ are two small subsets of DBpedia, and are both introduced in Chapter 5 Section 5.1.4.

Museum $\sqsubseteq$ Building

$\top \sqsubseteq \forall$.employer.Organisation

author(*Encyclopedia of Life*,*Smithsonian Institution*)

award(*James Dewar*,*Smithsonian Institution*)

doctoralAdvisor(*Thaddeus S. C. Lowe*,*Smithsonian Institution*)

MilitaryPerson(*Cher Ami*)

Building(*Saturn V Dynamic Test Vehicle*)

`occupation`(*Montgomery C. Meigs*,*Smithsonian Institution*)

`award`(*Joan Hill*,*Smithsonian Institution*)

`director`(*Museum of the Rockies*,*Smithsonian Institution*)

`occupation`(*Mohamed Al-Fayed*,*Hôtel Ritz Paris*)

`nationality`(*Mohamed Al-Fayed*,*Egyptians*)

`stateOfOrigin`(*Mohamed Al-Fayed*,*Egyptians*)

`birthPlace`(*Mohamed Al-Fayed*,*Egypt*)

Appendix C

Abbreviations

- DAG: Directed Acyclic Graph
- DL: Description Logics
- DUL: DOLCE+DnS Ultralite
- FOL: First Order Logic
- KB: Knowledge Base
- lemmaPOS: a lemma+part-of-speech pair, for instance "take$_V$" to designate an occurrence of the verb "to take", and "take$_N$" to designate an occurrence of the noun "take", regardless of their morphological inflections.
- NERC: Named Entity Recognition and Classification
- NLP: Natural Language Processing
- NNF: Negation Normal Form
- PMI: Pointwise Mutual Information

- PPMI: Positive Pointwise Mutual Information
- POS: Part-Of-Speech
- SW: Semantiwc Web

Appendix D

Bibliography

Bibliography

[ADML12] Sören Auer, Jan Demter, Michael Martin, and Jens Lehmann. LODStats–an extensible framework for high-performance dataset analytics. In *Knowledge Engineering and Knowledge Management*. Springer, 2012.

[AGM85] Carlos E Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The journal of symbolic logic*, 50(02), 1985.

[BB04] Marco Baroni and Silvia Bernardini. BootCaT: Bootstrapping Corpora and Terms from the Web. In *LREC proceedings*, 2004.

[BBFZ09] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226, 2009.

[BBL08] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the EL envelope further. In *Proceedings of the OWLED 2008 DC Workshop on OWL: Experiences and Directions*, 2008.

[BC99] Matthew Berland and Eugene Charniak. Finding parts in very large corpora. In *ACL proceedings*, 1999.

[BCD+93] Salem Benferhat, Claudette Cayrol, Didier Dubois, Jerome Lang, and Henri Prade. Inconsistency management and prioritized syntax-based entailment. In *IJCAI proceedings*, 1993.

[BCM+03] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The description logic handbook: theory, implementation and applications*. Cambridge university press, 2003.

[BEP+08] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD international conference on Management of data proceedings*, 2008.

[BGBA10] A.B. Benevides, G. Guizzardi, B.F.B. Braga, and J.P.A. Almeida. Validating modal aspects of OntoUML conceptual models using automatically generated visual world structures. *Journal of Universal Computer Science*, 16(20), 2010.

[BH95] Franz Baader and Bernhard Hollunder. Embedding defaults into terminological knowledge representation formalisms. *Journal of Automated Reasoning*, 14(1):149–180, 1995.

[BHBL09] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pages 205–227, 2009.

[BKLBW04] Salem Benferhat, Souhila Kaci, Daniel Le Berre, and Mary-Anne Williams. Weakening conflicting information for iterated revision and knowledge integration. *Artificial Intelligence*, 153(1), 2004.

[Bor85] Alexander Borgida. Language features for flexible handling of exceptions in information systems. *ACM Transactions on Database Systems (TODS)*, 10(4):565–603, 1985.

[BP10] Franz Baader and Rafael Peñaloza. Axiom pinpointing in general tableaux. *Journal of Logic and Computation*, 20(1):5–34, 2010.

[CAGV15a] Julien Corman, Nathalie Aussenac-Gilles, and Laure Vieu. Distributional semantics for ontology verification. In *StarSem proceedings*, 2015.

[CAGV15b] Julien Corman, Nathalie Aussenac-Gilles, and Laure Vieu. Ontological analysis for practical knowledge base debugging in Description Logics. In *CommonSense proceedings*, 2015.

[CAGV15c] Julien Corman, Nathalie Aussenac-Gilles, and Laure Vieu. Prioritized base debugging in Description Logics. In *Ontochange proceedings (IJCAI workshop)*, 2015.

[CAGV15d] Julien Corman, Nathalie Aussenac-Gilles, and Laure Vieu. Trimming a consistent OWL knowledge base, relying on linguistic evidence. In *LangAndOnto proceedings*, 2015.

[Cam81] Keith Campbell. The metaphysic of abstract particulars. *Midwest studies in philosophy*, 6(1):477–488, 1981.

[CDGL+05] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. DL-Lite: Tractable description logics for ontologies. In *AAAI proceedings*, 2005.

[CGHKS08] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Modular reuse of ontologies: Theory and practice. *Journal of Artificial Intelligence Research*, pages 273–318, 2008.

[Cim06] P. Cimiano. *Ontology learning and population from text: algorithms, evaluation and applications*. Springer, 2006.

[CKNZ10] Diego Calvanese, Evgeny Kharlamov, Werner Nutt, and Dmitriy Zheleznyakov. Evolution of DL- lite knowledge bases. In *ISWC proceedings*, 2010.

[Cla13] Stephen Clark. Vector Space Models of Lexical Meaning. In *Handbook of Contemporary Semantic Theory*. Blackwell, 2013.

[CRW13] Raphael Cóbe, Fillipe Resina, and Renata Wassermann. Merging Ontologies via Kernel Contraction. *ONTOBRAS proceedings*, pages 94–105, 2013.

[CV05] Philipp Cimiano and Johanna Völker. Towards large-scale, open-domain and ontology-based named entity classification. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, 2005.

[Dal88] Mukesh Dalal. Investigations into a theory of knowledge base revision: preliminary report. In *NCAI proceedings*, volume 2, pages 475–479, 1988.

[Del08] James P Delgrande. Horn Clause Belief Change: Contraction Functions. In *KR proceedings*, pages 156–165, 2008.

[DKSP10] Mariana Damova, Atanas Kiryakov, Kiril Simov, and Svetoslav Petrov. Mapping the central LOD ontologies to PROTON upper-level ontology. *Ontology Matching poceedings*, 2010.

[Doe03] Martin Doerr. The CIDOC conceptual reference module: an ontological approach to semantic interoperability of metadata. *AI magazine*, 24(3):75, 2003.

[ESo07] Jérôme Euzenat, Pavel Shvaiko, and others. *Ontology matching*, volume 333. Springer, 2007.

[FHP+06] Giorgos Flouris, Zhisheng Huang, Jeff Z Pan, Dimitris Plexousakis, and Holger Wache. Inconsistencies, negations and changes in ontologies. In *NCAI proceedings*, 2006.

[Flo06] Giorgos Flouris. *On Belief Change and Ontology Evolution*. PhD thesis, University of Crete, 2006.

[FR12] Sébastien Ferré and Sebastian Rudolph. Advocatus Diaboli–Exploratory Enrichment of Ontologies with Negative Constraints. *EKAW proceedings*, 2012.

[FS05] Gerhard Friedrich and Kostyantyn Shchekotykhin. A general diagnosis method for ontologies. In *ISWC proceedings*, 2005.

[GG95] Pierdaniele Giaretta and N Guarino. Ontologies and knowledge bases towards a terminological clarification. *Towards very large knowledge bases: knowledge building & knowledge sharing*, 25:32, 1995.

[GG08] Claudio Giuliano and Alfio Gliozzo. Instance-based ontology population exploiting named-entity substitution. In *COLING proceedings*, 2008.

[GKZ12] Bernardo Cuenca Grau, Evgeny Kharlamov, and Dmitriy Zheleznyakov. Ontology Contraction: Beyond the Propositional Paradise. In *AMW proceedings*, pages 62–74, 2012.

[GNP+12] Aldo Gangemi, Andrea Giovanni Nuzzolese, Valentina Presutti, Francesco Draicchio, Alberto Musetti, and Paolo Ciancarini. Automatic typing of DBpedia entities. In *ISWC proceedings*, 2012.

[GOS09] Nicola Guarino, Daniel Oberle, and Steffen Staab. What is an Ontology? In *Handbook on ontologies*, pages 1–17. Springer, 2009.

[Gru95] Thomas R Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5):907–928, 1995.

[GRV10] Birte Glimm, Sebastian Rudolph, and Johanna Völker. Integrated metamodeling and diagnosis in OWL 2. In *ISWC proceedings*, 2010.

[GSW89] Russell Greiner, Barbara A Smith, and Ralph W Wilkerson. A correction to the algorithm in Reiter's theory of diagnosis. *Artificial Intelligence*, 41(1):79–88, 1989.

[Gui05] Giancarlo Guizzardi. *Ontological foundations for structural conceptual models*. PhD thesis, CTIT, University of Twente, Netherlands, 2005.

[GW00] N. Guarino and C. Welty. A formal ontology of properties. In *EKAW proceedings*, 2000.

[GW02] Nicola Guarino and Christopher Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, 2002.

[GW09] Nicola Guarino and Christopher A Welty. An overview of OntoClean. In *Handbook on ontologies*, pages 201–220. Springer, 2009.

[Gä03] Peter Gärdenfors. *Belief revision*, volume 29. Cambridge University Press, 2003.

[Han91] Sven Ove Hansson. Belief contraction without recovery. *Studia Logica*, 50(2):251–260, 1991.

[Hea92] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *ACL proceedings*, 1992.

[HKR09] Pascal Hitzler, Markus Krotzsch, and Sebastian Rudolph. *Foundations of semantic web technologies*. CRC Press, 2009.

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible SROIQ. *KR proceedings*, 2006.

[Hor11] Matthew Horridge. *Justification based explanation in ontologies*. PhD thesis, the University of Manchester, 2011.

[JVZ+14] Elisabetta Jezek, Laure Vieu, Fabio Zanzotto, Guido Vetere, Alessandro Oltramari, Aldo Gangemi, and Rossella Varvara. Extending 'Senso Comune' with Semantic Role Sets. In Harry Bunt, editor, *Proceedings of the 10th Workshop on Interoperable Semantic Annotation (ISA-10)*, 2014.

[Kal06] Aditya Anand Kalyanpur. *Debugging and repair of owl ontologies*. PhD thesis, University of Maryland, 2006.

[KM89] Hirofumi Katsuno and Alberto O Mendelzon. A unified view of propositional knowledge base updates. In *Proceedings of the 11th international joint conference on Artificial intelligence-Volume 2*, pages 1413–1419, 1989.

[KPSH05] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James Hendler. Debugging unsatisfiable classes in OWL ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(4):268–293, 2005.

[Lam07] Sik Chun Joey Lam. *Methods for resolving inconsistencies in ontologies*. PhD thesis, University of Aberdeen, 2007.

[Mak87] David Makinson. On the status of the postulate of recovery in the logic of theory change. *Journal of Philosophical Logic*, 16(4):383–394, 1987.

[MBG+03] Claudio Masolo, Stefano Borgo, Aldo Gangemi, Nicola Guarino, and Alessandro Oltramari. WonderWeb Deliverable D18, Ontology Library. *LOA-ISTC, CNR, Tech. Rep*, 2003.

[Men91] Alberto O Mendelzon. On the difference between updating a knowledge base and revising it. In Hirofumi Katsuno, editor, *KR proceedings*, page 387, 1991.

[MGV+05] Claudio Masolo, Giancarlo Guizzardi, Laure Vieu, Emanuele Bottazzi, and Roberta Ferrario. Relational Roles and Qua-individuals. In Guido Boella, James Odell, Leendert van der Torre, and Harko Verhagen, editors, *Roles, an Interdisciplinary Perspective: Ontologies, Programming Languages, and Multiagent Systems. Papers from the AAAI Fall Symposium*, pages 103–112. AAAI Press, 2005.

[MJB12] Pablo N Mendes, Max Jakob, and Christian Bizer. DBpedia: A Multilingual Cross-domain Knowledge Base. In *LREC proceedings*, 2012.

[MLB05] Thomas Meyer, Kevin Lee, and Richard Booth. Knowledge integration for description logics. In *AAAI proceedings*, 2005.

[MLBP06] Thomas Meyer, Kevin Lee, Richard Booth, and Jeff Z Pan. Finding maximally satisfiable terminologies for the description logic ALC. In *NCAI proceedings*, 2006.

[MVB+04] Claudio Masolo, Laure Vieu, Emanuele Bottazzi, Carola Catenacci, Roberta Ferrario, Aldo Gangemi, and Nicola Guarino. Social Roles and their Descriptions. In *KR proceedings*, pages 267–277, 2004.

[Neb92] Bernhard Nebel. Syntax-based approaches to belief revision. *Belief revision*, 29:52–88, 1992.

[NVF11] Roberto Navigli, Paola Velardi, and Stefano Faralli. A graph-based algorithm for inducing lexical taxonomies from scratch. In *IJCAI*, 2011.

[PP06] Patrick Pantel and Marco Pennacchiotti. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *ACL proceedings*, 2006.

[PSK05] Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Debugging OWL ontologies. In *Proceedings of the 14th international conference on World Wide Web*, 2005.

[PVSFGP12] M. Poveda-Villalón, M. C. Suárez-Figueroa, and A. Gómez-Pérez. Did you validate your ontology? OOPS! In *ESWC proceedings*, 2012.

[QHH+08] Guilin Qi, Peter Haase, Zhisheng Huang, Qiu Ji, Jeff Z Pan, and Johanna Völker. A kernel revision operator for terminologies - algorithms and evaluation. In *ISWC proceedings*, 2008.

[QLB06] Guilin Qi, Weiru Liu, and David Bell. A revision-based approach to handling inconsistency in description logics. *Artificial Intelligence Review*, 26(1-2), 2006.

[RDH+04] A. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, and C. Wroe. OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns. *Engineering Knowledge in the Age of the Semantic Web*, pages 63–81, 2004.

[Rec03] Alan L Rector. Modularisation of domain ontologies implemented in description logics and related formalisms including OWL. In *Proceedings of the 2nd international conference on Knowledge capture*. ACM, 2003.

[Rei87] Raymond Reiter. A theory of diagnosis from first principles. *Artificial intelligence*, 32(1), 1987.

[Res97] P. Resnik. Selectional preference and sense disambiguation. In *ACL SIGLEX Workshop on Tagging Text with Lexical Semantics proceedings*, 1997.

[Rib13] Márcio Moretto Ribeiro. *Belief revision in non-classical logics*. Springer, 2013.

[RW08] Márcio Moretto Ribeiro and Renata Wassermann. The Ontology Reviser Plug-In for Protégé. In *WONTO proceedings*, 2008.

[RW09] Márcio M Ribeiro and Renata Wassermann. Base revision for ontology debugging. *Journal of Logic and Computation*, 19(5), 2009.

[RWFA13] Márcio M Ribeiro, Renata Wassermann, Giorgos Flouris, and Grigoris Antoniou. Minimal change: Relevance and recovery revisited. *Artificial Intelligence*, 201, 2013.

[RZ13] Catherine Roussey and Ondrej Zamazal. Antipattern Detection: How to Debug an Ontology without a Reasoner. In *WODOOM proceedings*, 2013.

[SB05] A. Schutz and P. Buitelaar. Relext: A tool for relation extraction from text in ontology extension. *ISWC proceedings*, 2005.

[SBG12] Tiago Prince Sales, Pedro Paulo Favato Barcelos, and Giancarlo Guizzardi. Identification of Semantic Anti-Patterns in Ontology-Driven Conceptual Modeling via Visual Simulation. In *FOIS proceedings*, 2012.

[SC03] Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *IJCAI proceedings*, 2003.

[Sch05] Stefan Schlobach. Diagnosing terminologies. In *AAAI proceedings*, 2005.

[SSZ09] Ulrike Sattler, Thomas Schneider, and Michael Zakharyaschev. Which kind of module should I extract? *Description Logics*, 477:78, 2009.

[TKM05] Ivan Terziev, Atanas Kiryakov, and Dimitar Manov. D. 1.8. 1 Base upper-level ontology (BULO) Guidance. *Deliverable of EU-IST Project IST*, 2005.

[TKMS03] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL proceedings*, 2003.

[TM08] Hristo Tanev and Bernardo Magnini. Weakly supervised approaches for ontology population. In *conference on Ontology Learning and Population proceedings*, 2008.

[TPo10] Peter D Turney, Patrick Pantel, and others. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188, 2010.

[VA07] Laure Vieu and Michel Aurnague. Part-of relations, functionality and dependence. *The categorization of spatial entities in language and cognition*, 20:307, 2007.

[VHC07] Johanna Völker, Pascal Hitzler, and Philipp Cimiano. Acquisition of OWL DL axioms from lexical resources. In *The Semantic Web: Research and Applications*, pages 670–685. Springer, 2007.

[VVSH07] Johanna Völker, Denny Vrandečić, York Sure, and Andreas Hotho. Learning disjointness. In *The Semantic Web: Research and Applications*, pages 175–189. Springer, 2007.

[VVSH08] Johanna Völker, Denny Vrandečić, York Sure, and Andreas Hotho. AEON–An approach to the automatic evaluation of ontologies. *Applied Ontology*, 3(1), 2008.

[Was00] Renata Wassermann. An algorithm for belief revision. In *KR proceedings*, 2000.

[Wel06] Chris Welty. OntOWLClean: Cleaning OWL ontologies with OWL. *Frontiers in artificial intelligence and applications*, 150:347, 2006.

[Win88] Marianne Winslett. Reasoning about action using a possible models approach. *Urbana*, 51:61801, 1988.