A logic-based framework for ontology comparison and module extraction in *DL-Lite*

Roman Kontchakov[a,*], Frank Wolter[b], Michael Zakharyaschev[a]

[a] *School of Computer Science and Information Systems, Birkbeck College London, U.K.*
[b] *Department of Computer Science, University of Liverpool, U.K.*

Abstract

We develop a formal framework for comparing different versions of *DL-Lite* ontologies. The main feature of our approach is that we take into account the vocabulary (= signature) with respect to which one wants to compare ontologies. Five variants of difference and inseparability relations between ontologies are introduced and their respective applications for ontology development and maintenance discussed. These variants are obtained by generalising the notion of conservative extension from mathematical logic and by distinguishing between differences that can be observed among concept inclusions, answers to queries over ABoxes, by taking into account additional context ontologies, and by considering a model-theoretic, language-independent notion of difference. We compare these variants, study their meta-properties, determine the computational complexity of the corresponding reasoning tasks, and present decision algorithms. Moreover, we show that checking inseparability can be automated by means of encoding into QBF satisfiability and using off-the-shelf general purpose QBF solvers.

Inseparability relations between ontologies are then used to develop a formal framework for (minimal) module extraction. We demonstrate that different types of minimal modules induced by these inseparability relations can be automatically extracted from real-world medium-size *DL-Lite* ontologies by composing the tractable syntactic locality-based module extraction algorithm with non-tractable extraction algorithms using the multi-engine QBF solver AQME. Finally, we explore the relationship between uniform interpolation (or forgetting) and inseparability between ontologies.

Key words: Description logic, ontology, module extraction, entailment, computational complexity, uniform interpolation, forgetting.

Contents

1 Introduction **2**

2 The *DL-Lite* class of description logics **5**

3 What is the difference? **8**

4 Semantic criteria of Σ-entailment **14**
4.1 Semantic criteria for $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ 14
4.2 Semantic criteria for $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ 16
4.3 Σ-difference 17

*Corresponding author
Email addresses: roman@dcs.bbk.ac.uk (Roman Kontchakov), frank@csc.liv.ac.uk (Frank Wolter), michael@dcs.bbk.ac.uk (Michael Zakharyaschev)

5 Robustness properties **18**
5.1 Robustness under definitorial extensions 18
5.2 Robustness under vocabulary extensions 19
5.3 Robustness under joins 19
5.4 Robustness under language extensions 20

6 Complexity of Σ-entailment **20**
6.1 Lower bounds 20
6.2 Complexity of language-dependent Σ-entailments 21
6.3 Decidability of Σ-model entailment 24

7 Inseparability modules **27**

8 Forgetting and uniform interpolation **30**

9 Experimental results **33**
9.1 QBF encodings 33
9.2 Experiments with Σ-entailment 34
9.3 Practical module extraction 36

10 Conclusion **38**

A Appendix **38**
A.1 Preliminaries 38
A.2 Proofs of results from Section 4 40
A.3 Proofs of results from Section 5 43
A.4 Proofs of results from Section 6 44
A.5 Proofs of results from Section 8 51

1. Introduction

In computer science, ontologies are used to provide a common vocabulary (or, in logic parlance, signature) for a domain of interest, together with a description of certain relationships between terms built from the vocabulary. Ontology languages based on description logics represent ontologies as 'TBoxes' (terminological boxes) containing inclusions between complex concepts over the vocabulary [1]. An increasingly important application of ontologies is management of large amounts of data, where ontologies are used to provide flexible and efficient access to repositories consisting of data sets of instances of concepts and relations. In description logics, such repositories are typically modelled as 'ABoxes' (assertion boxes) [1].

Developing and maintaining ontologies for this and other purposes is a rather difficult task. When using description logics (including the description logic based dialects of the Web Ontology Language OWL[1]), the ontology designer is supported by efficient reasoning tools for classification, instance checking and a variety of other reasoning tasks. However, this support is generally recognised to be insufficient when ontologies are developed not as 'monolithic entities' but by means of importing, merging, combining, refining and extending already existing ontologies. In all those cases, reasoning support for analysing the impact of the respective operation on the ontology would be extremely useful. Typical examples of such 'unorthodox' reasoning services include the following:

Comparing versions of ontologies. The standard syntactic `diff` utility is an indispensable tool for comparing different versions of text files, and it would be very helpful to have a similar versioning tool for ontologies. However, a purely syntactic operation of computing the difference between ontologies is of little value [2] because our concern now is not the syntactic form of the axioms, but their differing logical consequences. Moreover, instead

[1] `http://www.w3.org/2007/OWL/`

of comparing arbitrary logical consequences, it is more useful and informative to compare logical consequences over the *common vocabulary* Σ of the versions, or even such consequences regarding a certain subject matter corresponding to some subvocabulary of Σ. Thus, the reasoning service we need in this case should be able to compare the logical consequences of different versions of ontologies over some vocabulary Σ.

Ontology refinement. When refining an ontology by adding new axioms, one usually wants to preserve the relationships between terms of a certain part Σ of its vocabulary. The reasoning service required in such a case is to check whether the refined ontology has precisely the same logical consequences over Σ as the original one.

Ontology re-use. When importing an ontology, one wants to *use* its vocabulary Σ as originally defined. However, relationships between terms over Σ may change due to interaction with some axioms in the importing ontology. So again we need a reasoning service capable of checking whether new logical consequences over Σ are derivable (this service has been termed *safety checking* in [3]).

In all these and many other cases, we are interested in comparing logical consequences over some vocabulary Σ that can be drawn from two different ontologies. This gives rise to the three main notions we investigate in this paper: Σ-*difference*, Σ-*entailment*, and Σ-*inseparability*. Roughly, the Σ-difference between two ontologies is the set of 'formulas' over Σ that are derivable from one ontology but not from the other; one ontology Σ-entails another one if all Σ-formulas derivable from the latter are also derivable from the former; and two ontologies are Σ-inseparable if they Σ-entail each other.

In the discussion so far, we have not specified the language from which the logical consequences over Σ are drawn. This language depends on the application. For example, if one is mainly interested in terminological reasoning and differences visible in applications that use relationships between concepts, then an appropriate language is the set of all concept inclusions. The Σ-difference then consists of all concept inclusions over Σ derivable from one ontology but not from the other. And one ontology Σ-entails another ontology if every concept inclusion over Σ derivable from the latter is derivable from the former. If, however, one is mainly interested in using ontologies to query instance data, then it is more appropriate to consider a language for consequences over Σ that reflects, in some way, answers to queries in the signature Σ (or Σ-queries) over instance data in Σ. In this case, two ontologies should be Σ-inseparable if, and only if, they give the same answers to every Σ-query for any instance data over Σ. Even this language may be insufficient for applications where different versions of ontologies are imported into a context ontology, in which case two ontologies should be deemed Σ-inseparable only if *after* importing them into another ontology over Σ, the resulting extensions still give the same answers to Σ-queries.

The *first aim* of this paper is to give precise formalisations of five variants of Σ-difference, Σ-entailment and Σ-inseparability for ontologies given in the *DL-Lite* logics $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ and $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$. These variants of Σ-difference and Σ-entailment are obtained by distinguishing between differences visible among concept inclusions, answers to queries over ABoxes, by taking additional context ontologies into account, and by considering model-theoretic, language-independent notions of Σ-difference and Σ-entailment.

The *DL-Lite* family of description logics [4, 5, 6, 7] has been originally designed with the aim of providing query access to large amounts of data via a high-level conceptual (ontological) interface. Thus, the *DL-Lite* logics result from various compromises between (i) the necessity of retaining the data complexity of query answering as close as possible to the complexity of standard database query evaluation and (ii) the desire of having the expressive means for representing various constraints of data modelling formalisms such as the ER model and UML class diagrams [8]. For example, the logic $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ [9] (containing many other *DL-Lite* logics) can express is-a hierarchies of concepts, disjointness and covering constraints for concepts, and domain, range and cardinality constraints for binary relations. Instance checking in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ is in AC^0 for data complexity (i.e., of the same complexity as database query evaluation); however, answering conjunctive queries is coNP-complete. On the other hand, $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ cannot express covering constraints, but boasts AC^0 query answering (under the unique name assumption) [10]. To simplify presentation, in this paper we do not consider *DL-Lite* logics with role inclusions, focusing mainly on the impact of the Boolean constructs in concept inclusions as well as number restrictions. We also note that the *DL-Lite* family forms the basis of OWL 2 QL, one of the three profiles of the Web Ontology Language OWL 2.

For Σ-entailment and Σ-inseparability to be applicable in practice, one has to understand their basic meta-properties and develop corresponding decision algorithms. The important meta-properties of Σ-entailment to be formalised and investigated below specify the type of modifications of the signature Σ under which Σ-entailment is preserved, the

operations on TBoxes as well as the type of context ontologies that preserve Σ-entailment. Thus, the *second aim* of this paper is to compare our notions of Σ-difference and Σ-entailment, study their meta-properties, determine the computational complexity of deciding Σ-entailment and Σ-inseparability between *DL-Lite* ontologies, and develop decision algorithms.

The notions of Σ-entailment and Σ-inseparability investigated in this paper can be employed to provide a formal foundation for *module extraction* and *forgetting*.

Module extraction—the problem of finding a (minimal) subset of a given ontology that provides the same description of the relationships between terms over a given sub-vocabulary as the whole ontology—has recently become an active research topic; see, e.g., the recent volume on ontology modularisation [11] and the WoMO workshop series devoted to this problem [12, 13]. The reasons for this are manifold, with one of the most important being ontology re-use. It is often impossible and not even desirable to develop an entirely new ontology for every new application; a better methodology is to re-use appropriate existing ontologies. However, typically only a relatively small part of the vocabulary of a possibly large ontology is required, that is, one only needs a subset, or module, of the ontology that gives the same description of this sub-vocabulary. The phrase 'gives the same description of the vocabulary' is rather vague. It has been interpreted in a variety of ways, ranging from structural approaches [14, 15] to logic-based approaches [16, 17, 18]. It should not come as a surprise now that in this paper we propose to understand the claim that 'two ontologies give the same description of the vocabulary Σ' as 'the two ontologies are Σ-inseparable' in one of the senses described above. Thus, different variants of Σ-inseparability give rise to different modules and module extraction problems, and we use the notion of Σ-inseparability to develop a framework for investigating such modules and algorithms for their extraction.

Forgetting—the problem of constructing, given an ontology and a vocabulary Γ, a new ontology that results from the original one by 'forgetting' Γ but retaining all the information about the remaining symbols (that are not in Γ)—has been introduced and investigated in AI [19, 20, 21] and, under the name of uniform interpolation, in mathematical logic [22, 23, 24, 25]. Forgetting is of interest to ontology engineering for a variety of reasons [26, 27]. For example, similarly to module extraction it can be used to 'extract' from a given ontology another ontology that 'provides the same description of a certain vocabulary as the original one.' However, in contrast to module extraction,

– the new ontology has to be formulated without using the 'forgotten' symbols in Γ, and

– the concept inclusions of the new ontology do not necessarily come from the original one.

In this paper, we propose to define an ontology $\mathcal{O}_\Gamma$ to be a result of forgetting a vocabulary Γ in a given ontology $\mathcal{O}$ if $\mathcal{O}_\Gamma$ does not use any symbols from Γ and $\mathcal{O}$ and $\mathcal{O}_\Gamma$ are $\overline{\Gamma}$-inseparable for the vocabulary $\overline{\Gamma}$ that consists of all remaining (i.e., non-Γ) symbols in $\mathcal{O}$. So, like in the case of modules, different variants of Σ-inseparability induce different variants of forgetting.

Thus, the *third aim* of this paper is to give formal definitions of modules, module extraction, and forgetting using Σ-inseparability. We develop generic module extraction algorithms, which extract minimal modules using the algorithms deciding Σ-inseparability as oracles. We also present first results on forgetting and uniform interpolation.

Finally, our *fourth aim* is to find out whether the logic-based approach to detecting inseparability relations between *DL-Lite* ontologies can be used in practice, in particular, for minimal module extraction. With this aim in mind, we have conducted a series of experiments with a number of 'real-world' medium-size *DL-Lite*$^{\mathcal{N}}_{bool}$ ontologies (containing up to 1250 axioms). Instead of implementing dedicated algorithms for checking Σ-entailment, we have encoded the semantic criteria of Σ-entailment to be developed in this paper by means of quantified Boolean formulas (QBFs, for short) and then employed standard *off-the-shelf general purpose* QBF solvers governed by the self-adaptive multi-engine QBF solver AQME [28].

The paper, which is an extended version [29] (containing also results of [30]), is structured in the following way. We begin, in Section 2, by introducing the *DL-Lite* logics, discussing their properties we need in this paper and giving an illustrative example of a *DL-Lite*$^{\mathcal{N}}_{bool}$ ontology. In Section 3, we introduce, motivate and illustrate five different variants of Σ-entailment and its derivatives, Σ-difference and Σ-inseparability. We also start discussing the relationships between these variants. In Section 4, we formulate semantic criteria for Σ-entailment. We introduce and illustrate all the technical notions involved, but move the actual proofs to the appendix (apart from those that can be used for illustrative purposes). In Section 5, we investigate the important 'robustness' meta-properties of Σ-entailment mentioned above. In Section 6, we determine the computational complexity of deciding our Σ-entailment

relations between *DL-Lite* ontologies and present corresponding decision algorithms. Again, almost all the technical proofs can be found in the appendix. In Section 7, we show how the notion of Σ-inseparability can be employed to define modules, analyse relationships between modules, and design module extraction algorithms, while in Section 8, we discuss the notion of forgetting. In Section 9, we describe our experiments and analyse their results. We draw conclusions and discuss open problems and further directions of research in Section 10, which is followed by the technical appendix.

2. The *DL-Lite* class of description logics

One of the most interesting and promising recent applications of description logics (DLs, for short) is to provide access to large amounts of data through a high-level conceptual interface, which can be used in such areas as data integration and ontology-based data access. The reasoning services required in this context include the traditional knowledge base satisfiability and instance checking, as well as answering complex database-like queries by taking into account both the terminological axioms and the data stored in the knowledge base. As the amount of data is supposed to be large, the key property for this approach to be viable in practice is the efficiency of query evaluation, with the ideal target being traditional database query processing. With this aim in mind the *DL-Lite family* of DLs has been designed in [4, 5, 6, 7] and a supporting QuOnto system has been implemented [31, 32]. The *DL-Lite* family forms the basis of OWL 2 QL, one of the three profiles of OWL 2.[2] According to the current version of the official W3C profiles document, the purpose of OWL 2 QL is to be the language of choice for applications that use very large amounts of data and where query answering is the most important reasoning task. A detailed analysis of the impact of various DL constructs on the computational behaviour of *DL-Lite* logics has been conducted in [10], which resulted in a fine-grained classification of a more extensive class of *DL-Lite* related logics.

Two contradicting requirements have determined the shape of *DL-Lite* logics:

(i) answering conjunctive queries should be reducible to standard query evaluation in databases (in other words, it should belong to the complexity class AC^0 with respect to *data complexity*), and

(ii) the logics should be able to capture as much of typical conceptual modelling formalisms such as UML class diagrams and ER models as possible.

Before defining the syntax and semantics of *DL-Lite* logics formally, let us consider the UML class diagram depicted in Fig. 1 and representing (a portion of) a computer science department information system. For example, according to this diagram, research and visiting staff are disjoint, project managers can only be from the visiting and academic staff, each project is managed by one or two managers, and each researcher works on at least one project, and the other way round. A crucial observation here is that the information about binary relations such as 'manages' or 'works on' provided by the UML class diagram concerns only their domains and ranges (the domain of 'manages' is a subset of all project managers, while its range is the set of all projects) as well as multiplicity (each project is managed by at most two managers). This observation motivates the following description logic called $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ in [10] (and $\textit{DL-Lite}_{bool}$ in [9]).

The alphabet of $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ consists of three countably infinite sets: *object names* $a_1, a_2, \ldots$, *concept names* $A_1, A_2, \ldots$, and *role names* $P_1, P_2, \ldots$. Complex *roles* R and *concepts* C of $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ are defined inductively as follows:

$$
\begin{array}{lcl}
R & ::= & P_i \mid P_i^-, \\
B & ::= & \bot \mid \top \mid A_i \mid \geq q\, R, \\
C & ::= & B \mid \neg C \mid C_1 \sqcap C_2,
\end{array}
$$

where q is a positive integer (represented in binary). The concepts of the form B are called *basic*. Other standard concept constructs such as $\exists R$, $\leq q\, R$ and $C_1 \sqcup C_2$ can be introduced as abbreviations: $\exists R$ for $\geq 1\, R$, $\leq q\, R$ for $\neg(\geq q+1\, R)$,

[2] The OWL 2 profiles are fragments of the full OWL 2 that have been designed and standardised for specific application requirements; see `http://www.w3.org/TR/owl2-profiles/`.

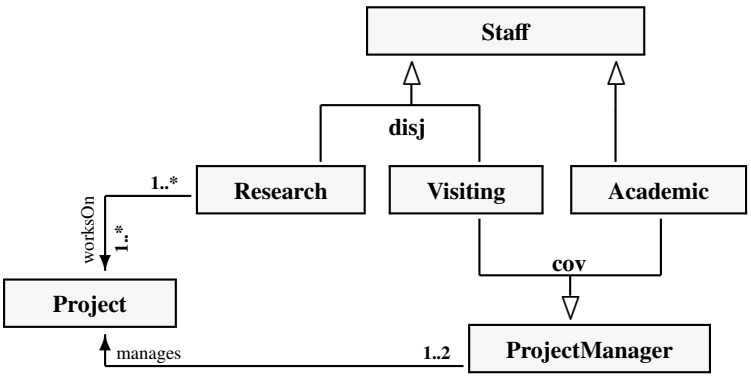

Figure 1: A UML class diagram.

and $C_1 \sqcup C_2$ for $\neg(\neg C_1 \sqcap \neg C_2)$. Concepts of the form $\leq q\,R$ and $\geq q\,R$ will be called *number restrictions*, and those of the form $\exists R$ and $\geq 1\,R$ *existential concepts*.

A *concept inclusion* in *DL-Lite*$^{\mathcal{N}}_{bool}$ is of the form $C_1 \sqsubseteq C_2$, where C_1 and C_2 are *DL-Lite*$^{\mathcal{N}}_{bool}$ concepts. A *TBox in DL-Lite*$^{\mathcal{N}}_{bool}$, denoted $\mathcal{T}$, is a finite set of concept inclusions in *DL-Lite*$^{\mathcal{N}}_{bool}$. As usual, we write $C_1 \equiv C_2$ instead of the two inclusions $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$.

We use $\ell(C)$ to denote the *length* of a concept C—i.e., the number of symbols required to write it down. The *length* (or *size*) $\ell(\mathcal{T})$ of a TBox $\mathcal{T}$ is defined by taking $\sum_{C \sqsubseteq D \in \mathcal{T}} (\ell(C) + \ell(D))$.

Example 1. The UML class diagram in Fig. 1 can be represented by the following *DL-Lite*$^{\mathcal{N}}_{bool}$ TBox:

$$\begin{aligned}
\exists\,\mathsf{manages} &\sqsubseteq \mathsf{ProjectManager}, & \exists\,\mathsf{worksOn} &\sqsubseteq \mathsf{Research},\\
\exists\,\mathsf{manages}^{-} &\sqsubseteq \mathsf{Project}, & \exists\,\mathsf{worksOn}^{-} &\sqsubseteq \mathsf{Project},\\
\mathsf{Project} &\sqsubseteq \exists\,\mathsf{manages}^{-}, & \mathsf{Research} &\sqsubseteq \exists\,\mathsf{worksOn},\\
\geq 3\ \mathsf{manages}^{-} &\sqsubseteq \bot, & \mathsf{Project} &\sqsubseteq \exists\,\mathsf{worksOn}^{-},\\
\mathsf{Research} &\sqsubseteq \mathsf{Staff}, & \mathsf{Visiting} &\sqsubseteq \mathsf{Staff},\\
\mathsf{Research} \sqcap \mathsf{Visiting} &\sqsubseteq \bot, & \mathsf{Academic} &\sqsubseteq \mathsf{Staff},\\
\mathsf{Visiting} &\sqsubseteq \mathsf{ProjectManager}, & \mathsf{Academic} &\sqsubseteq \mathsf{ProjectManager},\\
\mathsf{ProjectManager} &\sqsubseteq \mathsf{Academic} \sqcup \mathsf{Visiting}.
\end{aligned}$$

We will also consider a sub-language *DL-Lite*$^{\mathcal{N}}_{horn}$ of *DL-Lite*$^{\mathcal{N}}_{bool}$, called the *Horn fragment* of *DL-Lite*$^{\mathcal{N}}_{bool}$. The concept inclusions in *DL-Lite*$^{\mathcal{N}}_{horn}$ are restricted to the form

$$B_1 \sqcap \cdots \sqcap B_k \ \sqsubseteq\ B, \tag{Horn}$$

where B and the B_i are basic concepts. Note that the inclusions $\bigsqcap_k B_k \sqsubseteq \bot$ and $\top \sqsubseteq B$ are legal in *DL-Lite*$^{\mathcal{N}}_{horn}$. A *TBox in DL-Lite*$^{\mathcal{N}}_{horn}$ is a finite set of concept inclusions in *DL-Lite*$^{\mathcal{N}}_{horn}$. In the context of this fragment of *DL-Lite*$^{\mathcal{N}}_{bool}$, basic concepts will also be called *DL-Lite*$^{\mathcal{N}}_{horn}$ *concepts*. It is worth noting that in *DL-Lite*$^{\mathcal{N}}_{horn}$ we can express both *global functionality* of a role and *local functionality* (i.e., functionality restricted to a (basic) concept B) by means of the concept inclusions $\geq 2\,R \sqsubseteq \bot$ and $\geq 2\,R \sqcap B \sqsubseteq \bot$.

Let $\mathcal{L}$ be one of the languages *DL-Lite*$^{\mathcal{N}}_{bool}$ or *DL-Lite*$^{\mathcal{N}}_{horn}$. An *ABox* in $\mathcal{L}$, denoted $\mathcal{A}$, is a finite set of assertions of the form $C(a_i)$, $R(a_i, a_j)$, $a_i = a_j$ and $a_i \neq a_j$, where C is an $\mathcal{L}$-concept, R a role, and a_i, a_j are object names. An $\mathcal{L}$ *knowledge base* ($\mathcal{L}$-*KB*, for short) is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ with a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ both in $\mathcal{L}$.

By a *signature* we understand any *finite* set Σ of concept and role names. (As TBoxes in *DL-Lite*$^{\mathcal{N}}_{bool}$ do not contain object names, we do not have to include them in signatures.) Given a concept, role, TBox, ABox, or any other expression E in the alphabet of *DL-Lite*$^{\mathcal{N}}_{bool}$, we denote by $sig(E)$ the *signature of* E, that is, the set of concept and role names that occur in E. It is to be noted that $\bot$ and $\top$ are regarded as logical symbols, and so $sig(\bot) = sig(\top) = \emptyset$.

A concept (role, TBox, ABox, etc.) E is called a Σ-*concept* (*role*, *TBox*, *ABox*, etc., respectively) if $sig(E) \subseteq \Sigma$. Thus, P^- is a Σ-role if, and only if, $P \in \Sigma$.

Given a signature Σ, we define a Σ-*interpretation* $\mathcal{I}$ as a structure of the form $(\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a nonempty set, the *domain* of interpretation, and $\cdot^\mathcal{I}$ is an *interpretation function* that assigns to each concept name $A_i \in \Sigma$ a subset $A_i^\mathcal{I} \subseteq \Delta^\mathcal{I}$ of the domain, to each role name $P_i \in \Sigma$ a binary relation $P_i^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ over the domain, and to each object name a_i an element $a_i^\mathcal{I} \in \Delta^\mathcal{I}$. If $\mathcal{I}$ interprets all concept and role names or Σ is understood, then we usually drop the modifier Σ and call $\mathcal{I}$ simply an *interpretation*. For an interpretation $\mathcal{I}$ and a signature Σ, we denote by $\mathcal{I}\restriction_\Sigma$ the Σ-*reduct of* $\mathcal{I}$ *to* Σ, that is, the Σ-interpretation with domain $\Delta^\mathcal{I}$ in which $A_i^{\mathcal{I}\restriction_\Sigma} = A_i^\mathcal{I}$, for all concept names $A_i \in \Sigma$, $P_i^{\mathcal{I}\restriction_\Sigma} = P_i^\mathcal{I}$, for all role names $P_i \in \Sigma$, and $a_i^{\mathcal{I}\restriction_\Sigma} = a_i^\mathcal{I}$, for all object names a_i.

Complex roles and concepts are interpreted in $\mathcal{I}$ as follows:

$$
\begin{array}{rcll}
(P_i^-)^\mathcal{I} & = & \{(y,x) \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid (x,y) \in P_i^\mathcal{I}\}, & \text{(inverse role)} \\
\top^\mathcal{I} & = & \Delta^\mathcal{I}, & \text{(the whole domain)} \\
\bot^\mathcal{I} & = & \emptyset, & \text{(the empty set)} \\
(\geq q\, R)^\mathcal{I} & = & \{x \in \Delta^\mathcal{I} \mid \sharp\{y \in \Delta^\mathcal{I} \mid (x,y) \in R^\mathcal{I}\} \geq q\}, & \text{(at least } q \text{ } R\text{-successors)} \\
(\neg C)^\mathcal{I} & = & \Delta^\mathcal{I} \setminus C^\mathcal{I}, & \text{(not in } C\text{)} \\
(C_1 \sqcap C_2)^\mathcal{I} & = & C_1^\mathcal{I} \cap C_2^\mathcal{I}, & \text{(both in } C_1 \text{ and } C_2\text{)}
\end{array}
$$

where, for typographical reasons, we denote the cardinality of X by $\sharp X$ instead of the usual $|X|$.

The *satisfaction relation* $\models$ is defined by taking:

$$
\begin{array}{rcl}
\mathcal{I} \models C_1 \sqsubseteq C_2 & \text{iff} & C_1^\mathcal{I} \subseteq C_2^\mathcal{I}, \\
\mathcal{I} \models C(a_i) & \text{iff} & a_i^\mathcal{I} \in C^\mathcal{I}, \\
\mathcal{I} \models R(a_i, a_j) & \text{iff} & (a_i^\mathcal{I}, a_j^\mathcal{I}) \in R^\mathcal{I}, \\
\mathcal{I} \models a_i = a_j & \text{iff} & a_i^\mathcal{I} = a_j^\mathcal{I}, \\
\mathcal{I} \models a_i \neq a_j & \text{iff} & a_i^\mathcal{I} \neq a_j^\mathcal{I},
\end{array}
$$

where C, C_1, C_2 are $\mathcal{L}$-concepts, R a role, and a_i, a_j object names. An $\mathcal{L}$-KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is said to be *satisfiable* (or *consistent*) if there is an interpretation $\mathcal{I}$ satisfying all the members of $\mathcal{T}$ and $\mathcal{A}$. In this case we write $\mathcal{I} \models \mathcal{K}$ (as well as $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$) and say that $\mathcal{I}$ is a *model of* $\mathcal{K}$ (and of $\mathcal{T}$ and $\mathcal{A}$). A concept inclusion $C_1 \sqsubseteq C_2$ *follows from* (or is a *logical consequence of*) $\mathcal{T}$, $\mathcal{T} \models C_1 \sqsubseteq C_2$ in symbols, if every model of $\mathcal{T}$ satisfies $C_1 \sqsubseteq C_2$. A concept C is $\mathcal{T}$*-satisfiable* if there exists a model $\mathcal{I}$ of $\mathcal{T}$ with $C^\mathcal{I} \neq \emptyset$.

An (*essentially positive*) *existential query* $\mathsf{q}(x_1, \ldots, x_n)$ in $\mathcal{L}$ (or simply a *query*, if $\mathcal{L}$ is understood) is a first-order formula

$$\exists y_1 \ldots \exists y_m\, \varphi(x_1, \ldots, x_n, y_1, \ldots, y_m),$$

where φ is constructed, using only $\wedge$ and $\vee$, from atoms of the form $C(t)$ and $R(t_1, t_2)$, with C being an $\mathcal{L}$-concept, R a role, and t_i being either an object name or a variable from the list $x_1, \ldots, x_n, y_1, \ldots, y_m$. The free variables of q are called *distinguished variables* of q and the bound ones *non-distinguished variables* of q. We write $\mathsf{q}(x_1, \ldots, x_n)$ for a query with distinguished variables $x_1, \ldots, x_n$. Given a query $\mathsf{q}(\boldsymbol{x})$ with $\boldsymbol{x} = x_1, \ldots, x_n$ and an n-tuple $\boldsymbol{a}$ of object names, we write $\mathsf{q}(\boldsymbol{a})$ for the result of replacing every occurrence of x_i in $\mathsf{q}(\boldsymbol{x})$ with the ith member of $\boldsymbol{a}$. Queries containing no distinguished variables are called *ground* or *Boolean*.

Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation. An *assignment* $\mathfrak{a}$ in $\Delta^\mathcal{I}$ is a function associating with every variable y an element $\mathfrak{a}(y)$ of $\Delta^\mathcal{I}$. We will use the following notation: $a_i^{\mathcal{I},\mathfrak{a}} = a_i^\mathcal{I}$ and $y^{\mathcal{I},\mathfrak{a}} = \mathfrak{a}(y)$. The *satisfaction relation* for

existential queries with respect to a given assignment $\mathfrak{a}$ is defined inductively by taking:

$$\begin{aligned}
\mathcal{I} \models^{\mathfrak{a}} C(t) \quad &\text{iff} \quad t^{\mathcal{I},\mathfrak{a}} \in C^{\mathcal{I}},\\
\mathcal{I} \models^{\mathfrak{a}} R(t_1,t_2) \quad &\text{iff} \quad (t_1^{\mathcal{I},\mathfrak{a}}, t_2^{\mathcal{I},\mathfrak{a}}) \in R^{\mathcal{I}},\\
\mathcal{I} \models^{\mathfrak{a}} \varphi_1 \wedge \varphi_2 \quad &\text{iff} \quad \mathcal{I} \models^{\mathfrak{a}} \varphi_1 \text{ and } \mathcal{I} \models^{\mathfrak{a}} \varphi_2,\\
\mathcal{I} \models^{\mathfrak{a}} \varphi_1 \vee \varphi_2 \quad &\text{iff} \quad \mathcal{I} \models^{\mathfrak{a}} \varphi_1 \text{ or } \mathcal{I} \models^{\mathfrak{a}} \varphi_2,\\
\mathcal{I} \models^{\mathfrak{a}} \exists y\, \varphi \quad &\text{iff} \quad \mathcal{I} \models^{\mathfrak{b}} \varphi, \text{ for some assignment } \mathfrak{b} \text{ in } \Delta^{\mathcal{I}} \text{ that may differ from } \mathfrak{a} \text{ on } y.
\end{aligned}$$

For a ground query $\mathsf{q}(\boldsymbol{a})$, the satisfaction relation does not depend on the assignment $\mathfrak{a}$; so we write $\mathcal{I} \models \mathsf{q}(\boldsymbol{a})$ instead of $\mathcal{I} \models^{\mathfrak{a}} \mathsf{q}(\boldsymbol{a})$.

For a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, we say that a tuple $\boldsymbol{a}$ of object names from $\mathcal{A}$ is a *certain answer* to $\mathsf{q}(\boldsymbol{x})$ with respect to $\mathcal{K}$ and write $\mathcal{K} \models \mathsf{q}(\boldsymbol{a})$, if $\mathcal{I} \models \mathsf{q}(\boldsymbol{a})$ whenever $\mathcal{I} \models \mathcal{K}$. A certain answer to a ground query $\mathsf{q}(\boldsymbol{a})$ with respect to $\mathcal{K}$ is either 'yes' if $\mathcal{K} \models \mathsf{q}(\boldsymbol{a})$ and 'no' otherwise. The *query answering problem* in $\mathcal{L}$ can be formulated as follows: given an $\mathcal{L}$-KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, a query $\mathsf{q}(\boldsymbol{x})$ in $\mathcal{L}$, and a tuple $\boldsymbol{a}$ of object names from $\mathcal{A}$, decide whether $\mathcal{K} \models \mathsf{q}(\boldsymbol{a})$.

Remark 2. The reader must have probably noticed that the class of essentially positive existential queries in $\mathcal{L}$ we deal with in this paper is larger than the standard class of *positive existential queries* which can be built, using $\wedge$ and $\vee$, only from atoms of the form $A_i(t)$ and $P_j(t_1, t_2)$ where the A_i and P_j are concept and role names, respectively. In particular, in the case of *DL-Lite*$_{bool}^{\mathcal{N}}$, essentially positive existential queries may contain 'complex atoms' $C(t)$ like $(\neg(\geq 7\, P_j^-) \wedge \neg A_i)(y)$. The reason why we consider more complex queries will be discussed in Section 5.1. Note, however, that query answering for essentially positive existential queries in $\mathcal{L}$ can be reduced to query answering in $\mathcal{L}$ using positive existential queries: given an $\mathcal{L}$-KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and an essentially positive existential query $\mathsf{q}(\boldsymbol{x})$ in $\mathcal{L}$, one can replace every occurrence of a complex atom $C(t)$ in $\mathsf{q}(\boldsymbol{x})$ with $A_C(t)$, for a fresh concept name A_C, and add to $\mathcal{T}$ the definition $A_C \equiv C$, which clearly belongs to $\mathcal{L}$. Denote the resulting positive existential query by $\mathsf{q}'(\boldsymbol{x})$ and the resulting $\mathcal{L}$-KB by $\mathcal{K}'$. It is readily seen that, for every tuple $\boldsymbol{a}$ of object names from $\mathcal{A}$, we have $\mathcal{K} \models \mathsf{q}(\boldsymbol{a})$ if, and only if, $\mathcal{K}' \models \mathsf{q}'(\boldsymbol{a})$.

Remark 3 (on the unique name assumption). According to the definitions given above, we do not adopt here the *unique name assumption* (UNA, for short), which can be formulated as follows. We say that an interpretation $\mathcal{I}$ is a *model of a KB* $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ *under the UNA* if $\mathcal{I} \models \mathcal{K}$ and $a_i^{\mathcal{I}} \neq a_j^{\mathcal{I}}$, for any distinct object names a_i and a_j *occurring in* $\mathcal{A}$. Instead, we follow the more liberal approach taken in OWL: the UNA is dropped, but the user is provided with means, = and ≠, to say explicitly which object names must denote the same individual and which must be different. Of course, we can always enforce the UNA by adding to each ABox $\mathcal{A}$ the inequalities $a_i \neq a_j$ for all pairs of distinct object names a_i and a_j occurring in $\mathcal{A}$. In fact, we shall see in Theorem 18 that for our purposes it does not matter which of the two approaches is taken. However, the complexity of standard reasoning tasks like satisfiability checking or query answering in the *DL-Lite* logics does depend on whether the UNA is adopted or not. We recall the following complexity results from [9, 10] for our *DL-Lite* logics with and without the UNA:

With the UNA: the satisfiability problem for knowledge bases is NP-complete for *DL-Lite*$_{bool}^{\mathcal{N}}$ and P-complete for *DL-Lite*$_{horn}^{\mathcal{N}}$ with respect to *combined complexity*; answering existential queries is in AC^0 for *DL-Lite*$_{horn}^{\mathcal{N}}$ KBs and coNP-complete for *DL-Lite*$_{bool}^{\mathcal{N}}$ KBs with respect to *data complexity*.

Without the UNA: satisfiability is NP-complete and query answering is coNP-complete for both *DL-Lite*$_{bool}^{\mathcal{N}}$ and *DL-Lite*$_{horn}^{\mathcal{N}}$; by limiting number restrictions to global functionality constraints $\geq 2\, R \sqsubseteq \bot$ and existential concepts $\exists R$ only, we reduce the complexity of satisfiability and query answering for the Horn fragment to P; and if the functionality constraints are also removed then the complexity of satisfiability becomes the same as in the UNA case, while query answering for the Horn fragment drops to LogSpace (or even to AC^0 if the use of = is not allowed).

3. What is the difference?

In this section, we give precise definitions of various notions of difference, entailment, and inseparability between ontologies with respect to a signature and discuss how these notions are related to each other.

Intuitively, an ontology $\mathcal{T}_1$ is inseparable from an ontology $\mathcal{T}_2$ with respect to a signature Σ if $\mathcal{T}_1$ and $\mathcal{T}_2$ cannot be distinguished from each other by means of their consequences over Σ. To make this intuition precise, we have to specify a language from which the consequences are drawn. As we consider ontologies formulated in the *DL-Lite* logics $\mathcal{L} = \textit{DL-Lite}^{\mathcal{N}}_{bool}, \textit{DL-Lite}^{\mathcal{N}}_{horn}$, the most obvious language for consequences is probably the concept inclusions in $\mathcal{L}$. Thus, we can say that TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\mathcal{L}$ are Σ-inseparable if $\mathcal{T}_1$ and $\mathcal{T}_2$ imply the same Σ-concept inclusions in $\mathcal{L}$. The corresponding non-symmetric notion of Σ-entailment is formulated as follows: $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ if every Σ-concept inclusion in $\mathcal{L}$ that follows from $\mathcal{T}_2$ also follows from $\mathcal{T}_1$ (so $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-inseparable if, and only if, they Σ-entail each other). Finally, the Σ-difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ can be defined as the set of all Σ-concept inclusions in $\mathcal{L}$ that follow from $\mathcal{T}_2$ but not from $\mathcal{T}_1$. To indicate that we are interested in consequences of ontologies in the form of concept inclusions, we prefix these notions of difference, entailment and inseparability with the modifier *concept*. Here is a formal definition:

Definition 4. The Σ-*concept difference between TBoxes* $\mathcal{T}_1$ *and* $\mathcal{T}_2$ *in* $\mathcal{L}$ is the set $\mathsf{cDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ of all Σ-concept inclusions $C \sqsubseteq D$ in $\mathcal{L}$ such that $\mathcal{T}_1 \not\models C \sqsubseteq D$ and $\mathcal{T}_2 \models C \sqsubseteq D$.

$\mathcal{T}_1$ Σ-*concept entails* $\mathcal{T}_2$ *in* $\mathcal{L}$ if $\mathsf{cDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-*concept inseparable in* $\mathcal{L}$ if they Σ-concept entail each other in $\mathcal{L}$.

Σ-concept inseparability between $\mathcal{T}_1$ and $\mathcal{T}_2$ means that $\mathcal{T}_1$ can be replaced by $\mathcal{T}_2$ in any application that is only concerned with Σ-concept inclusions in $\mathcal{L}$ (we elaborate on this claim below). An ontology developer who wants to compare two versions $\mathcal{T}_1$ and $\mathcal{T}_2$ of an ontology with respect to a signature Σ can check whether they are Σ-concept inseparable and, if this is not the case, further inspect $\mathsf{cDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ and $\mathsf{cDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_2, \mathcal{T}_1)$ to analyse the Σ-differences between these versions.

Remark 5. The notion of Σ-concept entailment between TBoxes is a generalisation of the notion of conservative extension investigated in [24, 33] for expressive descriptions logics such as $\mathcal{ALC}$ and $\mathcal{ALCQI}$. Namely, a TBox $\mathcal{T}_2$ is a conservative extension of a TBox $\mathcal{T}_1$ if $\mathcal{T}_1 \subseteq \mathcal{T}_2$ and $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ for $\Sigma = \textit{sig}(\mathcal{T}_1)$. The notion of conservative extension originates from mathematical logic where it is used, e.g., for relative consistency proofs in arithmetic and set theory; see [34] for more information. In computer science, conservative extensions have found applications in modular software specification and verification [35, 36, 37, 38]. The first papers suggesting to use conservative extensions (or variants thereof) for modular ontology engineering were [39, 16, 24]. In answer set programming, modularity and variations of conservative extensions have been investigated in, e.g., [40, 41, 42, 43].

Concept inclusions are not the only interesting type of consequences of TBoxes. In the context of *DL-Lite* ontologies, answers to queries over ABoxes are probably of even greater importance than concept inclusions. The following example shows that the 'concept-based' notions of difference and entailment introduced above are not appropriate for applications that involve query answering. (The claims made in the examples below will be explained in a informal way; strict proofs can be easily given using the semantic criteria to be discussed in Section 4.)

Example 6. Let $\Sigma = \{\mathsf{Lecturer}, \mathsf{Course}\}$,

$$\mathcal{T}_1 = \emptyset \quad \text{and} \quad \mathcal{T}_2 = \{\mathsf{Lecturer} \sqsubseteq \exists \mathsf{teaches},\ \exists \mathsf{teaches}^- \sqsubseteq \mathsf{Course}\}.$$

Intuitively, the only (non-tautological) consequence of $\mathcal{T}_2$ over Σ is 'if there is a lecturer, then there is a course', which cannot be expressed by means of Σ-concept inclusions. Thus, $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-concept inseparable (in both $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and $\textit{DL-Lite}^{\mathcal{N}}_{horn}$). On the other hand, $\mathcal{T}_1$ and $\mathcal{T}_2$ become Σ-separable if they are used to query ABoxes. For instance, let $\mathcal{A} = \{\mathsf{Lecturer}(a)\}$ and $\mathsf{q} = \exists y\, \mathsf{Course}(y)$. Although both $\textit{sig}(\mathcal{A})$ and $\textit{sig}(\mathsf{q})$ are in Σ, they nevertheless separate $\mathcal{T}_1$ and $\mathcal{T}_2$ because $(\mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}$ but $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}$.

Thus, in applications where TBoxes are used to query ABoxes, $\mathcal{T}_1$ cannot be regarded as indistinguishable from $\mathcal{T}_2$ with respect to Σ because one can find a Σ-ABox and a Σ-query in the presence of which $\mathcal{T}_1$ behaves differently from $\mathcal{T}_2$.

To take into account the differences between TBoxes that can be detected by means of ABoxes and queries, we propose the following definition:

Definition 7. The Σ-*query difference between TBoxes* $\mathcal{T}_1$ *and* $\mathcal{T}_2$ *in* $\mathcal{L}$ is the set $\mathsf{qDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ of pairs of the form $(\mathcal{A}, \mathsf{q}(\boldsymbol{x}))$, where $\mathcal{A}$ is a Σ-ABox in $\mathcal{L}$ and $\mathsf{q}(\boldsymbol{x})$ a Σ-query in $\mathcal{L}$ such that $(\mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$ and $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$, for some tuple $\boldsymbol{a}$ of object names from $\mathcal{A}$.

$\mathcal{T}_1$ Σ-*query entails* $\mathcal{T}_2$ *in* $\mathcal{L}$ if $\mathsf{qDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-*query inseparable in* $\mathcal{L}$ if they Σ-query entail each other in $\mathcal{L}$.

In the definition of Σ-query difference, we take into consideration *arbitrary* Σ-ABoxes in $\mathcal{L}$. The reason for this is that, during the ontology design phase, the data repositories to which the ontology will be applied are often either completely unknown or are subject to more or less frequent changes. Thus, to assume that we have a fixed ABox is unrealistic when checking Σ-query differences between ontologies, and that is why in our approach we regard ABoxes as 'black boxes'. This notion of Σ-query difference and entailment has been discussed in [25] and investigated for the description logic $\mathcal{EL}$ in [44].

As we shall see later (cf. Theorem 24), for *DL-Lite*$^{\mathcal{N}}_{horn}$ TBoxes, Σ-concept entailment in *DL-Lite*$^{\mathcal{N}}_{horn}$ implies Σ-concept entailment in *DL-Lite*$^{\mathcal{N}}_{bool}$. However, this implication does not hold for Σ-query entailment, as shown by the following example:

Example 8. Let $\Sigma = \{\mathsf{Lecturer}\}$,

$$\mathcal{T}_1 = \emptyset \quad \text{and} \quad \mathcal{T}_2 = \{\mathsf{Lecturer} \sqsubseteq \exists \mathsf{teaches},\ \mathsf{Lecturer} \sqcap \exists \mathsf{teaches}^{-} \sqsubseteq \bot\}.$$

Then $\mathcal{T}_1$ does not Σ-query entail $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{bool}$. Indeed, for $\mathcal{A} = \{\mathsf{Lecturer}(a)\}$ and $\mathsf{q} = \exists y\, \neg \mathsf{Lecturer}(y)$, we have $(\mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}$ and $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}$. On the other hand, as we do not allow negation in *DL-Lite*$^{\mathcal{N}}_{horn}$ queries, one can show that $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{horn}$.

Similarly to Σ-concept inseparability, Σ-query inseparability between TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ means that $\mathcal{T}_1$ can be replaced by $\mathcal{T}_2$ in the applications where only answers to Σ-queries over Σ-ABoxes are of interest. However, this informal explanation should be taken with caution. To see why, recall that one of the reasons for studying inseparability and difference is ontology *re-use*: instead of constructing ontologies from scratch, it is often preferable to import (parts of) already existing ontologies. In other words, ontologies are designed as the union

$$\mathcal{T}_{self} \cup \mathcal{T}_{imp},$$

where $\mathcal{T}_{self}$ is an ontology developed specifically for the given application and $\mathcal{T}_{imp}$ is an imported ontology. A problem arises when we have a choice between different versions of such $\mathcal{T}_{imp}$ or when it is preferable to import only a small subset of $\mathcal{T}_{imp}$ (later, in Section 7, called a module) that contains all the relevant information for the new application. In these cases, we would like to be able to detect whether it makes any difference if we import a version $\mathcal{T}'_{imp}$ or a version $\mathcal{T}''_{imp}$ of $\mathcal{T}_{imp}$ and, likewise, whether it makes any difference if $\mathcal{T}_{imp}$ itself is imported or only its subset $\mathcal{M}$. In other words, we would like to know whether

- $\mathcal{T}_{self} \cup \mathcal{T}'_{imp}$ and $\mathcal{T}_{self} \cup \mathcal{T}''_{imp}$ are Σ-inseparable, and whether
- $\mathcal{T}_{self} \cup \mathcal{M}$ and $\mathcal{T}_{self} \cup \mathcal{T}_{imp}$ are Σ-inseparable,

where Σ is the signature required for the application. Now, instead of checking Σ-inseparability *after* taking the union with $\mathcal{T}_{self}$, it would be much more useful to be able to check Σ-inseparability *independently* of $\mathcal{T}_{self}$ and *before* importing the ontologies we are interested in. Consider, for example, a situation when $\mathcal{T}_{self}$ is still evolving or subject to frequent changes. Thus, it would be desirable to have a notion of Σ-inseparability with the following *replacement property*:

(replace) if $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-inseparable in $\mathcal{L}$, then $\mathcal{T} \cup \mathcal{T}_1$ and $\mathcal{T} \cup \mathcal{T}_2$ are Σ-inseparable in $\mathcal{L}$, for all Σ-TBoxes $\mathcal{T}$ in $\mathcal{L}$.

If a notion of Σ-inseparability has this property, then Σ-inseparability of $\mathcal{T}_1$ and $\mathcal{T}_2$ ensures that $\mathcal{T}_1$ can be replaced by $\mathcal{T}_2$ within any context Σ-TBox $\mathcal{T}$ in the given language $\mathcal{L}$. For further discussions of the replacement property, we refer the reader to Section 7, where Σ-inseparability is used for module extraction, and to Section 5.4, where we consider context TBoxes that are given in expressive DLs such as $\mathcal{SHIQ}$.

Unfortunately, not all the notions of inseparability introduced so far enjoy the replacement property.

Example 9. Let again $\mathcal{T}_1 = \emptyset$ and $\mathcal{T}_2$ be the TBox from Example 8 saying that every lecturer teaches and that a lecturer is not something that is taught. As before, consider $\Sigma = \{\mathsf{Lecturer}\}$. Then $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-concept inseparable in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$. But for $\mathcal{T} = \{\top \sqsubseteq \mathsf{Lecturer}\}$, we have $\mathcal{T}_1 \cup \mathcal{T} \not\models \top \sqsubseteq \bot$ and $\mathcal{T}_2 \cup \mathcal{T} \models \top \sqsubseteq \bot$, i.e., the former TBox is consistent while the latter is not. Thus, no difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ is visible if we only consider Σ-concept inclusions; that the two TBoxes are indeed different becomes apparent in the presence of the extra Σ-TBox $\mathcal{T}$.

To take such context ontologies into account, we introduce two stronger variants of Σ-inseparability that, by their very definitions, enjoy the replacement property.

Definition 10. *The strong Σ-concept difference in $\mathcal{L}$ between TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$* is the set $\mathsf{scDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ of pairs $(\mathcal{T}, C \sqsubseteq D)$ where $\mathcal{T}$ is a Σ-TBox in $\mathcal{L}$ and $C \sqsubseteq D$ belongs to $\mathsf{cDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T} \cup \mathcal{T}_1, \mathcal{T} \cup \mathcal{T}_2)$. $\mathcal{T}_1$ *strongly Σ-concept entails $\mathcal{T}_2$ in $\mathcal{L}$* if $\mathsf{scDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. $\mathcal{T}_1$ and $\mathcal{T}_2$ are *strongly Σ-concept inseparable in $\mathcal{L}$* if they strongly Σ-concept entail each other in $\mathcal{L}$.

The strong Σ-query difference in $\mathcal{L}$ between $\mathcal{T}_1$ and $\mathcal{T}_2$ is the set $\mathsf{sqDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ of triples $(\mathcal{T}, \mathcal{A}, \mathsf{q}(\boldsymbol{x}))$ such that $\mathcal{T}$ is a Σ-TBox in $\mathcal{L}$ and $(\mathcal{A}, \mathsf{q}(\boldsymbol{x})) \in \mathsf{qDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T} \cup \mathcal{T}_1, \mathcal{T} \cup \mathcal{T}_2)$. $\mathcal{T}_1$ *strongly Σ-query entails $\mathcal{T}_2$ in $\mathcal{L}$* if $\mathsf{sqDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. $\mathcal{T}_1$ and $\mathcal{T}_2$ are *strongly Σ-query inseparable in $\mathcal{L}$* if they strongly Σ-query entail each other in $\mathcal{L}$.

Thus, if two versions of ontologies are strongly Σ-inseparable for their shared signature Σ, then they can be safely replaced by each other within any ontology $\mathcal{T}$ which only uses symbols from Σ; after such a replacement no differences between the sets of derivable Σ-concept inclusions (or answers to Σ-queries) can be detected. In the context of defining modules within ontologies, taking into account changes to ontologies and context ontologies has been strongly advocated in [3], which inspired our definition; see also Section 7.

The notions of difference and inseparability introduced so far are *language-dependent* because the set of syntactic objects collected in the difference between two ontologies depends on the description logic under consideration. We have already seen in Example 8 that Σ-query entailment in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$ does not coincide with Σ-query entailment in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$, even for $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ TBoxes. Here is another example showing that strong Σ-concept entailment in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ does not imply strong Σ-concept entailment in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$.

Example 11. Consider the $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ TBoxes

$$\mathcal{T}_1 = \{\mathsf{Male} \sqcap \mathsf{Female} \sqsubseteq \bot,\ \top \sqsubseteq \exists\, \mathsf{father},\ \top \sqsubseteq \exists\, \mathsf{mother},\ \exists\, \mathsf{father}^- \sqsubseteq \mathsf{Male},\ \exists\, \mathsf{mother}^- \sqsubseteq \mathsf{Female}\},$$
$$\mathcal{T}_2 = \{\top \sqsubseteq \exists\, \mathsf{id},\ \mathsf{Male} \sqcap \exists\, \mathsf{id}^- \sqsubseteq \bot,\ \mathsf{Female} \sqcap \exists\, \mathsf{id}^- \sqsubseteq \bot\},$$

and let $\Sigma = \{\mathsf{Male}, \mathsf{Female}, \mathsf{father}, \mathsf{mother}\}$. It follows from $\mathcal{T}_2$ that the range of the role id is disjoint from Male and Female. Now let $\mathcal{T} = \{\top \sqsubseteq \mathsf{Male} \sqcup \mathsf{Female}\}$. Then $\mathcal{T} \cup \mathcal{T}_1$ is consistent, but $\mathcal{T} \cup \mathcal{T}_2$ is inconsistent. Thus we have $\mathcal{T} \cup \mathcal{T}_2 \models \top \sqsubseteq \bot$, while $\mathcal{T} \cup \mathcal{T}_1 \not\models \top \sqsubseteq \bot$, and so $\mathcal{T}_1$ does not strongly Σ-concept entail $\mathcal{T}_2$ in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$. However, one can show that $\mathcal{T}_2$ is strongly Σ-concept entailed by $\mathcal{T}_1$ in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$. Intuitively, the reason for this is that in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ we cannot express that Male and Female together cover the whole domain.

Language-dependence of the notions of difference between ontologies is unproblematic and justified if the languages involved in the application are known in advance. For example, if the application involves conceptual reasoning in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ or $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$, or query answering over ABoxes in these languages, the corresponding notions introduced above are entirely appropriate. Moreover, if weaker descriptions logics or query languages than the ones considered above are used, it is still *sound* to work with the notions of difference introduced so far as no relevant differences are missed. In some cases, however, one might be interested in importing *DL-Lite* ontologies into ontologies formulated in more expressive languages such as $\mathcal{SHIQ}$ [1] or even first-order logic. Or one might be interested in querying *DL-Lite* ontologies in more expressive languages than essentially positive existential queries. In these cases, our notions of difference can be incomplete because more expressive languages can potentially detect differences that are not observable in *DL-Lite*. The following example illustrates this point.

Example 12. Let $\mathcal{T}_1 = \emptyset, \mathcal{T}_2 = \{\top \sqsubseteq (\geq 2P)\}$ and $\Sigma = \emptyset$. The only difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ with respect to the empty signature Σ is that $\mathcal{T}_1$ has a model with domain of cardinality one, but $\mathcal{T}_2$ does not have such a model. Using this observation, one can show that $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ for all the notions of Σ-entailment introduced above. However, the first-order Σ-sentence $\varphi = \exists x \exists y\, (x \neq y)$ distinguishes between $\mathcal{T}_1$ and $\mathcal{T}_2$ since $\mathcal{T}_1 \not\models \varphi$ and $\mathcal{T}_2 \models \varphi$.

Instead of defining and investigating Σ-entailment for other languages such as $\mathcal{SHIQ}$ or first-order logic, in this paper we consider a language-independent, purely model-theoretic notion of difference which covers all the differences detectable in standard description logics, first-order and even second-order logic. Apart from that, we will show in Section 5 that strong Σ-query entailment in *DL-Lite*$^{\mathcal{N}}_{bool}$ is actually extremely robust in terms of language extensions within the family of description logics (see Theorem 39).

Definition 13. The Σ-*model difference between TBoxes* $\mathcal{T}_1$ *and* $\mathcal{T}_2$ is the class $\mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of all Σ-interpretations $\mathcal{I}$ for which there exists a model $\mathcal{I}_1$ of $\mathcal{T}_1$ with $\mathcal{I}_1 \restriction_\Sigma = \mathcal{I}$ but there is no model $\mathcal{I}_2$ of $\mathcal{T}_2$ with $\mathcal{I}_2 \restriction_\Sigma = \mathcal{I}$. We say that $\mathcal{T}_1$ Σ-*model entails* $\mathcal{T}_2$ if $\mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-*model inseparable* if they Σ-model entail each other.

Observe that, for $\mathcal{T}_1$, $\mathcal{T}_2$ and Σ from Example 12, $\mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ consists of all isomorphic copies of the Σ-interpretation whose domain has exactly one element. We give one more example illustrating our language-independent notion of Σ-difference.

Example 14. Let $\mathcal{T}_1$ be a TBox in *DL-Lite*$^{\mathcal{N}}_{horn}$ stating, using an auxiliary role name R, that concept B is nonempty:

$$\mathcal{T}_1 = \{\, \top \sqsubseteq \exists R,\ \exists R^- \sqsubseteq B \,\}.$$

Consider also a TBox $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{horn}$ stating that P is an injective function from A to B:

$$\mathcal{T}_2 = \{\, A \equiv \exists P,\ \exists P^- \sqsubseteq B,\ \geq 2\, P \sqsubseteq \bot,\ \geq 2\, P^- \sqsubseteq \bot \,\}.$$

Let $\Sigma = \{A, B\}$. Then $\mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ is the set of Σ-interpretations $\mathcal{I}$ in which $B^\mathcal{I}$ is nonempty and the cardinality of $A^\mathcal{I}$ is larger than the cardinality of $B^\mathcal{I}$ (and so there cannot be an injection from $A^\mathcal{I}$ to $B^\mathcal{I}$). As this set of interpretations is nonempty, $\mathcal{T}_1$ does not Σ-model entail $\mathcal{T}_2$. One can show, however, that $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ for all the language-dependent notions of Σ-entailment introduced above (see Example 22 below).

The following proposition provides some basic implications between the variants of Σ-entailment introduced above; a systematic investigation will be conducted in the next section.

Proposition 15. *Let* $\mathcal{L}$ *be one of DL-Lite*$^{\mathcal{N}}_{bool}$ *or DL-Lite*$^{\mathcal{N}}_{horn}$, $\mathcal{T}_1$ *and* $\mathcal{T}_2$ *TBoxes in* $\mathcal{L}$, *and* Σ *a signature.*

(i) *If* $\mathcal{T}_1$ Σ*-query entails* $\mathcal{T}_2$ *in* $\mathcal{L}$ *then* $\mathcal{T}_1$ Σ*-concept entails* $\mathcal{T}_2$ *in* $\mathcal{L}$. *In other words, if* $\mathsf{cDiff}^\mathcal{L}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \neq \emptyset$ *then* $\mathsf{qDiff}^\mathcal{L}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \neq \emptyset$.

(ii) *If* $\mathcal{T}_1$ Σ*-model entails* $\mathcal{T}_2$ *then* $\mathcal{T}_1$ *strongly* Σ*-query entails* $\mathcal{T}_2$ *in* $\mathcal{L}$. *In other words, if* $\mathsf{sqDiff}^\mathcal{L}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \neq \emptyset$ *then* $\mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \neq \emptyset$.

Proof. (i) To see that any difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ detectable by means of concept inclusions can also be detected by means of queries, suppose that we have $\mathcal{T}_1 \not\models C_1 \sqsubseteq C_2$ and $\mathcal{T}_2 \models C_1 \sqsubseteq C_2$, for some Σ-concept inclusion $C_1 \sqsubseteq C_2$ in $\mathcal{L}$. Consider the ABox $\mathcal{A} = \{C_1(a)\}$ and the query $\mathsf{q} = C_2(a)$. Then $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}$, while $(\mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}$. (Note that in *DL-Lite*$^{\mathcal{N}}_{horn}$ both the ABox and the query are defined correctly as $C_1 = B_1 \sqcap \dots \sqcap B_k$ and $C_2 = B$, where $B, B_1, \dots, B_k$ are basic concepts, and so $\mathcal{A} = \{B_1(a), \dots, B_k(a)\}$ and $\mathsf{q} = B(a)$.)

(ii) To see that any difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ detectable by triples $(\mathcal{T}, \mathcal{A}, \mathsf{q}(\boldsymbol{x}))$ can also be detected by means of Σ-interpretations, suppose that $(\mathcal{T} \cup \mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$ and $(\mathcal{T} \cup \mathcal{T}_2, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$, where $\mathcal{T}$, $\mathcal{A}$ and q contain symbols from Σ only. Take a model $\mathcal{I}$ of $(\mathcal{T} \cup \mathcal{T}_1, \mathcal{A})$ such that $\mathcal{I} \not\models \mathsf{q}(\boldsymbol{a})$. We show that $\mathcal{I} \restriction_\Sigma \in \mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. Indeed, otherwise we would have a model $\mathcal{I}'$ of $\mathcal{T}_2$ such that $\mathcal{I} \restriction_\Sigma = \mathcal{I}' \restriction_\Sigma$. But then, since $\mathcal{T}$, $\mathcal{A}$ and q use symbols from Σ only, $\mathcal{I}'$ would also be a model of $(\mathcal{T} \cup \mathcal{T}_2, \mathcal{A})$ and $\mathcal{I}' \not\models \mathsf{q}(\boldsymbol{a})$, contrary to $(\mathcal{T} \cup \mathcal{T}_2, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$. ❑

We conclude this section with two important observations. First we consider Σ-entailment between *DL-Lite* TBoxes containing no role names—in essence, Σ-entailment between propositional theories—and show that in this case all the variants of Σ-entailment introduced above coincide. And then we prove that our notions of Σ-entailment do not depend on the unique name assumption (UNA), as promised in Section 2.

Example 16 (Σ-entailment in propositional logic). If a TBox $\mathcal{T}$ does not contain any role names then we can identify concept names with propositional variables and regard $\mathcal{T}$ as a finite set $\mathcal{T}^*$ of propositional (Boolean) formulas (with the obvious correspondence between the concept construct $\sqcap$ and Boolean conjunction $\wedge$ and between concept inclusion $\sqsubseteq$ and Boolean implication $\to$). Moreover, if $\mathcal{T}$ is a *DL-Lite*$^{\mathcal{N}}_{horn}$ TBox, then $\mathcal{T}^*$ is a finite set of propositional Horn formulas. This brings us to Σ-entailment between propositional theories.

A *propositional theory* is just a finite set of propositional formulas, and a *propositional signature* is just a set of propositional variables. Let Σ be such a signature. Say that a propositional theory Φ Σ-*entails* a propositional theory Ψ if, for every propositional formula φ over Σ, we have $\Phi \models \varphi$ whenever $\Psi \models \varphi$. This notion can be characterised in purely model-theoretic terms: Φ Σ-entails Ψ if, and only if, for every propositional model I (assigning truth-values to propositional variables) of Φ, there exists a propositional model I' of Ψ that coincides with I on the variables in Σ. Indeed, the implication ($\Leftarrow$) is trivial. To show the converse, suppose that Φ Σ-entails Ψ, but there is a model I of Φ such that no model of Ψ coincides with I on the variables from Σ. Consider the formula

$$\chi_{I,\Sigma} \quad = \quad \neg\Big(\bigwedge_{p\in\Sigma,\ I\models p} p \quad \wedge \quad \bigwedge_{p\in\Sigma,\ I\not\models p} \neg p \Big).$$

By our assumption, $\Psi \models \chi_{I,\Sigma}$. But then we must have $\Phi \models \chi_{I,\Sigma}$, contrary to $I \models \Phi$ and $I \not\models \chi_{I,\Sigma}$.

It is also well-known that a propositional Horn theory Φ Σ-entails a propositional Horn theory Ψ if, and only if, for every *Horn formula* φ over Σ, $\Psi \models \varphi$ implies $\Phi \models \varphi$. In other words, for Horn theories it does not make any difference whether one considers Horn or arbitrary formulas in the language-dependent notion of Σ-entailment (cf. Theorem 24 below).

Thus, in contrast to the notions of Σ-entailment between *DL-Lite* TBoxes, in the propositional case the canonical language-dependent notion of Σ-entailment coincides with the model-theoretic notion of Σ-entailment.

Theorem 17. *Let $\mathcal{L}$ be one of DL-Lite*$^{\mathcal{N}}_{bool}$ *or DL-Lite*$^{\mathcal{N}}_{horn}$*. Let $\mathcal{T}_1$, $\mathcal{T}_2$ be TBoxes in $\mathcal{L}$ without occurrences of role names and Σ a signature. Then the following conditions are equivalent*:

- $\mathcal{T}_1$ (*strongly*) Σ-*concept entails* $\mathcal{T}_2$ *in* $\mathcal{L}$;
- $\mathcal{T}_1$ (*strongly*) Σ-*query entails* $\mathcal{T}_2$ *in* $\mathcal{L}$;
- $\mathcal{T}_1$ Σ-*model entails* $\mathcal{T}_2$;
- $\mathcal{T}_1^*$ Σ-*entails* $\mathcal{T}_2^*$ (*as propositional theories*).

Proof. Since $\mathcal{T}_1$ and $\mathcal{T}_2$ do not contain role names, we may assume without loss of generality (see Theorem 32 below) that Σ contains no role names. It should be clear from Example 16 that $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in $\mathcal{L}$ if, and only if, $\mathcal{T}_1^*$ Σ-entails $\mathcal{T}_2^*$. Thus, in view of Proposition 15, it suffices to show that if $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in $\mathcal{L}$ then $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$. Suppose otherwise. Then there is a model $\mathcal{I}$ of $\mathcal{T}_1$ such that for no model $\mathcal{I}'$ of $\mathcal{T}_2$ do we have $\mathcal{I} \restriction_\Sigma = \mathcal{I}' \restriction_\Sigma$. In fact, as $\mathcal{T}_1$ and $\mathcal{T}_2$ contain no role names, we can find such an $\mathcal{I}$ whose domain consists of a single point, say x. Then, similarly to the argument in Example 16, we take the Σ-concept inclusion $\top \sqsubseteq C_{\mathcal{I},\Sigma}$, where

$$C_{\mathcal{I},\Sigma} \quad = \quad \neg\Big(\bigsqcap_{A_i\in\Sigma,\ x\in A_i^{\mathcal{I}}} A_i \quad \sqcap \quad \bigsqcap_{A_i\in\Sigma,\ x\notin A_i^{\mathcal{I}}} \neg A_i \Big).$$

By our assumption, $\mathcal{T}_2 \models \top \sqsubseteq C_{\mathcal{I},\Sigma}$, and so $\mathcal{T}_1 \models \top \sqsubseteq C_{\mathcal{I},\Sigma}$, contrary to $\mathcal{I} \models \mathcal{T}_1$ and $\mathcal{I} \not\models \top \sqsubseteq C_{\mathcal{I},\Sigma}$. ❑

As mentioned in Section 2, there are two main paradigms for interpreting object names. One of them (typically adopted in the DL community) treats different object names from a given ABox as denoting different objects in interpretations; it is known as the unique name assumption (UNA). According to the other paradigm (which is standard in the OWL community as well as in first-order logic), no assumption is made as to how object names can be interpreted in general, but the users are provided with the ABox constructs = and ≠ in order to impose any constraints on object name interpretations they want. For example, to simulate the UNA, we can add to the ABoxes we are interested in the inequalities $a_i \neq a_j$ for all pairs of distinct object names a_i and a_j occurring in the ABoxes. Fortunately, in the context of the present investigation, it does not matter which of the two paradigms is adopted.

Theorem 18. *Let $\mathcal{L} \in \{\text{DL-Lite}^{\mathcal{N}}_{bool}, \text{DL-Lite}^{\mathcal{N}}_{horn}\}$. Let $\mathcal{T}_1$, $\mathcal{T}_2$ be TBoxes in $\mathcal{L}$ and Σ a signature. Then, for any variant of Σ-entailment introduced above, $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ in $\mathcal{L}$ under the UNA if, and only if, $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ in $\mathcal{L}$ without the UNA (but with $=$ and $\neq$).*

Proof. The claim is clear for Σ-concept, strong Σ-concept and Σ-model entailments because no ABoxes are involved in their definitions.

Consider Σ-query entailment. As was observed above, the case without the UNA covers the one with the UNA. So suppose that $\mathcal{T}_1$ does not Σ-query entail $\mathcal{T}_2$ without the UNA and show that $\mathcal{T}_1$ still does not Σ-query entail $\mathcal{T}_2$ under the UNA. Let $\mathcal{A}$ be a Σ-ABox in $\mathcal{L}$ and $\mathsf{q}(\boldsymbol{x})$ a Σ-query in $\mathcal{L}$ such that $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ but $(\mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$ for some tuple $\boldsymbol{a}$ from $\mathcal{A}$. Let $\mathcal{I}$ be a model of $(\mathcal{T}_1, \mathcal{A})$ (without the UNA) and $\mathfrak{a}$ an assignment with $\mathfrak{a}(x_i) = a_i$ such that $\mathcal{I} \not\models^{\mathfrak{a}} \mathsf{q}(\boldsymbol{x})$. Define an equivalence relation $\sim$ on the set of object names by taking $a_i \sim a_j$ if, and only if, $a_i^{\mathcal{I}} = a_j^{\mathcal{I}}$. Take a member a_ξ from each $\sim$-equivalence class ξ and define $\mathcal{A}'$ and $\mathsf{q}'(\boldsymbol{x})$ to be the ABox and query that result from $\mathcal{A}$ and $\mathsf{q}(\boldsymbol{x})$ by replacing every a_i with a_ξ for the $\sim$-equivalence class ξ of a_i. Then clearly $\mathcal{I}$ is a model of $(\mathcal{T}_1, \mathcal{A}')$ under the UNA and $\mathcal{I} \not\models \mathsf{q}'(\boldsymbol{a}')$, where $\boldsymbol{a}'$ is the tuple obtained from $\boldsymbol{a}$ by replacing every a_i with a_ξ for the $\sim$-equivalence class ξ of a_i. On the other hand, we immediately obtain from $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ that $\mathsf{q}'(\boldsymbol{a}')$ holds in any model of $(\mathcal{T}_2, \mathcal{A}')$ under the UNA. ❑

For technical reasons, it will be more convenient for us to adopt the UNA, or, which is the same, to assume that every ABox $\mathcal{A}$ contains inequalities $a_i \neq a_j$ for all distinct a_i, a_j occurring in $\mathcal{A}$.

4. Semantic criteria of Σ-entailment

In this section, we give semantic criteria for the language-dependent notions of Σ-entailment in $\text{DL-Lite}^{\mathcal{N}}_{bool}$ and $\text{DL-Lite}^{\mathcal{N}}_{horn}$. These criteria will be used to classify the notions of Σ-entailment, investigate their robustness properties in Section 5, provide tight complexity bounds for deciding Σ-entailment in Section 6, and design practical decision procedures in Section 9. Detailed proofs of all the results are given in the appendix (Section A.2).

4.1. Semantic criteria for $\text{DL-Lite}^{\mathcal{N}}_{bool}$

According to Proposition 15, Σ-model entailment implies all the language-dependent variants of Σ-entailment considered in this paper. Thus, to develop model-theoretic characterisations of language-dependent notions of Σ-entailment, we have to weaken the following condition characterising Σ-model entailment between TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$:

(model) every model of $\mathcal{T}_1$ can be transformed into a model of $\mathcal{T}_2$ by changing the interpretation of non-Σ-symbols.

We will do this by means of additional modifications of models of $\mathcal{T}_1$ when transforming them into models of $\mathcal{T}_2$. Our criteria have a somewhat syntactic flavour in the sense that they are formulated in terms of *types*—syntactic abstractions of domain elements—realised in models. The advantage of such characterisations is that they can be used directly for designing decision algorithms, despite the fact that the underlying models are often infinite, as neither $\text{DL-Lite}^{\mathcal{N}}_{bool}$ nor $\text{DL-Lite}^{\mathcal{N}}_{horn}$ has the finite model property [4]. Needless to say, however, that the correctness of the type-based characterisations presented below requires model constructions, which can be found in the technical appendix.

Let Σ be a signature and Q a set of positive natural numbers containing 1. A *basic* ΣQ*-concept* is any concept of the form $\bot$, $\top$, A_i, $\geq q\, R$, for some $A_i \in \Sigma$, Σ-role R and $q \in Q$, and by a ΣQ*-literal* we mean a basic ΣQ-concept or its negation. A ΣQ*-type* is a set $\boldsymbol{t}$ of ΣQ-literals containing $\top$ and such that the following conditions hold:

- for every ΣQ-literal C, either $C \in \boldsymbol{t}$ or $\neg C \in \boldsymbol{t}$,
- if the numbers $q < q'$ are both in Q and $\geq q'\, R \in \boldsymbol{t}$ then $\geq q\, R \in \boldsymbol{t}$,
- if the numbers $q < q'$ are both in Q and $\neg(\geq q\, R) \in \boldsymbol{t}$ then $\neg(\geq q'\, R) \in \boldsymbol{t}$.

Clearly, for every interpretation $\mathcal{I}$ and every point $x \in \Delta^{\mathcal{I}}$, the set

$$\boldsymbol{t}_{\mathcal{I}}(x) = \{C \mid x \in C^{\mathcal{I}},\ C \text{ a } \Sigma Q\text{-literal}\} \tag{1}$$

is a ΣQ-type. Conversely, for each ΣQ-type $\boldsymbol{t}$ with $\bot \notin \boldsymbol{t}$, there is an interpretation $\mathcal{I}$ with a point x such that $x \in C^{\mathcal{I}}$ for all $C \in \boldsymbol{t}$. In this case we say that $\boldsymbol{t}$ is *realised* (at x) in $\mathcal{I}$. Thus, ΣQ-types can indeed be regarded as abstractions of domain elements.

Definition 19. Given a TBox $\mathcal{T}$, we call a ΣQ-type $\mathcal{T}$*-realisable* if it is realised in a model of $\mathcal{T}$. A set Ξ of ΣQ-types is said to be $\mathcal{T}$*-realisable* if there is a model of $\mathcal{T}$ realising *all* the types from Ξ. We also say that Ξ is *precisely* $\mathcal{T}$*-realisable* if there is a model $\mathcal{I}$ of $\mathcal{T}$ realising all the types in Ξ, with every ΣQ-type realised in $\mathcal{I}$ being in Ξ.

Now, returning back to the characterisation **(model)** of Σ-model entailment, we see that if $\mathcal{I} \restriction_\Sigma = \mathcal{I}' \restriction_\Sigma$—i.e., $\mathcal{I}'$ is obtained from $\mathcal{I}$ by modifying the interpretation of non-Σ-symbols—then $\mathcal{I}$ and $\mathcal{I}'$ realise the same ΣQ-types, for every set Q of numerical parameters. Thus, if $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$ then

- every $\mathcal{T}_1$-realisable ΣQ-type is $\mathcal{T}_2$-realisable; moreover,
- every precisely $\mathcal{T}_1$-realisable set of ΣQ-types is precisely $\mathcal{T}_2$-realisable.

These two conditions are much more flexible than **(model)**: because of using types as abstractions of domain elements, the domain of the model is not fixed anymore, and so we can manipulate the domain elements by removing some of them or introducing new ones. The following two theorems state that these conditions indeed provide the semantic characterisations of the Σ-entailments we are looking for.

For a TBox $\mathcal{T}$, let $Q_{\mathcal{T}}$ denote the set of numerical parameters occurring in $\mathcal{T}$, together with number 1.

Theorem 20. *The following conditions are equivalent for TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$ and a signature Σ:*

($\mathbf{ce_b}$) *$\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$;*

(r) *every $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type is $\mathcal{T}_2$-realisable.*

Note that this equivalence is almost trivial if one considers $\Sigma\mathbb{N}$-types (i.e., types using arbitrary parameters) instead of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Thus, the message here is that it is sufficient to consider only the parameters from $Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$.

The next theorem characterises the remaining language-dependent variants of Σ-entailment for DL-Lite$^{\mathcal{N}}_{bool}$ TBoxes.

Theorem 21. *The following conditions are equivalent for TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$ and a signature Σ:*

($\mathbf{sce_b}$) *$\mathcal{T}_1$ strongly Σ-concept entails $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$;*

($\mathbf{qe_b}$) *$\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$;*

($\mathbf{sqe_b}$) *$\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$;*

(pr) *every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is precisely $\mathcal{T}_2$-realisable.*

Comparing these two criteria, we see that Σ-concept entailment is 'local' in the sense that it refers to a single point in a model, while Σ-query and strong Σ-concept/query entailments are 'global' because all points in a model have to be considered.

Example 22. To illustrate the criteria, we re-use Examples 6 and 14.

(i) Consider first the TBoxes $\mathcal{T}_1$, $\mathcal{T}_2$ and the signature $\Sigma = \{\mathsf{Lecturer}, \mathsf{Course}\}$ from Example 6. There are exactly four $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types: $\{\neg\mathsf{Lecturer}, \neg\mathsf{Course}\}$, $\{\mathsf{Lecturer}, \neg\mathsf{Course}\}$, $\{\neg\mathsf{Lecturer}, \mathsf{Course}\}$, $\{\mathsf{Lecturer}, \mathsf{Course}\}$, and all of them are $\mathcal{T}_1$-realisable. To see that $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ it remains to check that all these types are $\mathcal{T}_2$-realisable. On the other hand, the singleton $\{\{\mathsf{Lecturer}, \neg\mathsf{Course}\}\}$ is precisely $\mathcal{T}_1$-realisable but not precisely $\mathcal{T}_2$-realisable. Thus, $\mathcal{T}_1$ does not Σ-query entail $\mathcal{T}_2$.

(ii) Consider the TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ from Example 14: the former states that B is nonempty, while the latter that there is an injection from A to B. Let $\Sigma = \{A, B\}$. The four $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types $\{\neg A, \neg B\}, \{A, \neg B\}, \{\neg A, B\}, \{A, B\}$ are all $\mathcal{T}_i$-realisable, for $i = 1, 2$. Thus, $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$. To see that $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$, let Ξ be a precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Then there exists $\boldsymbol{t} \in \Xi$ with $B \in \boldsymbol{t}$. Take a Σ-interpretation $\mathcal{I}$ precisely realising Ξ and such that the set $\{d \in \Delta^{\mathcal{I}} \mid \boldsymbol{t} = \boldsymbol{t}_{\mathcal{I}}(d)\}$ is countably infinite, for every (of the at most four) $\boldsymbol{t} \in \Xi$. Then there exists an injection from $A^{\mathcal{I}}$ into $B^{\mathcal{I}}$ because $B^{\mathcal{I}}$ is countably infinite and $A^{\mathcal{I}}$ is either empty or countably infinite. Thus Ξ is precisely $\mathcal{T}_2$-realisable.

It is of interest to observe that all models $\mathcal{I}$ of $\mathcal{T}_2$ precisely realising $\Xi = \{\{A, \neg B\}, \{A, B\}\}$ are infinite, because $A^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $B^{\mathcal{I}}$ is a proper subset of $\Delta^{\mathcal{I}}$.

4.2. Semantic criteria for DL-Lite$^{\mathcal{N}}_{horn}$

The language of *DL-Lite*$^{\mathcal{N}}_{horn}$ does not contain negation; it operates only with basic concepts. Like in the previous section, we use the modifier ΣQ to indicate that a syntactic object is built up using concept and role names from Σ and numerical parameters from Q. For example, a *ΣQ-concept inclusion in DL-Lite*$^{\mathcal{N}}_{horn}$ is a concept inclusion of the form $B_1 \sqcap \cdots \sqcap B_k \sqsubseteq B$, where $B_1, \dots, B_k, B$ are basic ΣQ-concepts. As usual, the empty conjunction $\bigsqcap_{i \in \emptyset} B_i$ is understood as $\top$, which is a basic ΣQ-concept for any Σ and Q.

Given a ΣQ-type $\boldsymbol{t}$, we define its 'positive part' $\boldsymbol{t}^+$, which does not include negative literals, by taking:

$$\boldsymbol{t}^+ \;=\; \{B \in \boldsymbol{t} \mid B \text{ a basic concept}\}.$$

Say that a ΣQ-type $\boldsymbol{t}_1$ is *positively contained* in a ΣQ-type $\boldsymbol{t}_2$ if $\boldsymbol{t}_1^+ \subseteq \boldsymbol{t}_2^+$. Clearly, a ΣQ-type is uniquely determined by its positive part. Thus, we can (and frequently will) define a ΣQ-type $\boldsymbol{t}$ by giving only its positive part $\boldsymbol{t}^+$. Here is a first example of such a definition.

Given a TBox $\mathcal{T}$ in *DL-Lite*$^{\mathcal{N}}_{horn}$ and a ΣQ-type $\boldsymbol{t}$ with $\Sigma \subseteq sig(\mathcal{T})$ and $Q_{\mathcal{T}} \subseteq Q$, we define the *$\mathcal{T}$-closure of $\boldsymbol{t}$* to be the $sig(\mathcal{T})Q$-type, denoted $\mathsf{cl}_{\mathcal{T}}(\boldsymbol{t})$, in which $(\mathsf{cl}_{\mathcal{T}}(\boldsymbol{t}))^+$ consists of all basic $sig(\mathcal{T})Q$-concepts B such that

$$\mathcal{T} \models \bigsqcap_{B_k \in \boldsymbol{t}^+} B_k \sqsubseteq B.$$

It is well-known that $\mathsf{cl}_{\mathcal{T}}(\boldsymbol{t})$ can be computed in polynomial time in the size of $\mathcal{T}$. The following lemma provides a simple standard criterion for $\mathcal{T}$-realisability of types when $\mathcal{T}$ is a TBox in *DL-Lite*$^{\mathcal{N}}_{horn}$.

Proposition 23. *Let $\mathcal{T}$ be a TBox in DL-Lite*$^{\mathcal{N}}_{horn}$*. Then a ΣQ-type $\boldsymbol{t}$ is $\mathcal{T}$-realisable if, and only if, $\boldsymbol{t} = \mathsf{cl}_{\mathcal{T}}(\boldsymbol{t}) \restriction_{\Sigma}$ and $\bot \notin \boldsymbol{t}$.*

Here, for a $\Sigma' Q$-type $\boldsymbol{t}$, we denote by $\boldsymbol{t} \restriction_{\Sigma}$ the restriction of $\boldsymbol{t}$ to Σ-concepts, that is, $\boldsymbol{t} \restriction_{\Sigma} = \{C \in \boldsymbol{t} \mid C \text{ a } \Sigma Q\text{-literal}\}$.

Turning to type-based criteria for Σ-entailment in *DL-Lite*$^{\mathcal{N}}_{horn}$, we first observe that for Σ-concept entailment no new criterion is required because it coincides with Σ-concept entailment for *DL-Lite*$^{\mathcal{N}}_{bool}$. Thus, we generalise the well-known result from propositional logic according to which two propositional Horn theories entail the same Horn formulas if, and only if, these theories have the same consequences in the class of all propositional formulas.

Theorem 24. *For any TBoxes $\mathcal{T}_1$, $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$ *and any signature Σ, the following two conditions are equivalent*:

(ce$_{\mathbf{h}}$) *$\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$;

(ce$_{\mathbf{b}}$) *$\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{bool}$.

Proof. The implication **(ce$_{\mathbf{b}}$)** $\Rightarrow$ **(ce$_{\mathbf{h}}$)** is obvious. To show the converse, suppose that $\mathcal{T}_1$ does not Σ-concept entail $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{bool}$. Without loss of generality, we may assume that $\Sigma \subseteq sig(\mathcal{T}_1 \cup \mathcal{T}_2)$. If this is not the case, one can add $A \sqsubseteq A$ and $\exists P \sqsubseteq \exists P$ to, say, $\mathcal{T}_1$, for all $A, P \in \Sigma$ that are not in $sig(\mathcal{T}_1 \cup \mathcal{T}_2)$. By Theorem 20, there exists a $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type $\boldsymbol{t}$ that is not $\mathcal{T}_2$-realisable. Consider now the $\mathcal{T}_1$- and $\mathcal{T}_2$-closures $\mathsf{cl}_{\mathcal{T}_1}(\boldsymbol{t})$ and $\mathsf{cl}_{\mathcal{T}_2}(\boldsymbol{t})$ of $\boldsymbol{t}$. Since $\boldsymbol{t}$ is $\mathcal{T}_1$-realisable, we have $\mathsf{cl}_{\mathcal{T}_1}(\boldsymbol{t}) \restriction_{\Sigma} = \boldsymbol{t}$ by Proposition 23. On the other hand, as $\boldsymbol{t}$ is not $\mathcal{T}_2$-realisable, Proposition 23 means that $\boldsymbol{t}$ is properly positively contained in $\mathsf{cl}_{\mathcal{T}_2}(\boldsymbol{t}) \restriction_{\Sigma}$. Therefore, there is $B \in \mathsf{cl}_{\mathcal{T}_2}(\boldsymbol{t}) \restriction_{\Sigma} \setminus \mathsf{cl}_{\mathcal{T}_1}(\boldsymbol{t}) \restriction_{\Sigma}$ such that

$$\mathcal{T}_1 \not\models \bigsqcap_{B_k \in \boldsymbol{t}^+} B_k \sqsubseteq B \qquad \text{and} \qquad \mathcal{T}_2 \models \bigsqcap_{B_k \in \boldsymbol{t}^+} B_k \sqsubseteq B,$$

DL-Lite$^{\mathcal{N}}_{bool}$:	Σ-concept	$\Leftarrow$	Σ-query	$\Leftrightarrow$	strong Σ-concept	$\Leftrightarrow$	strong Σ-query	$\Leftarrow$	Σ-model
	$\Updownarrow$		$\Downarrow$		$\Downarrow$		$\Downarrow$		$\Updownarrow$
DL-Lite$^{\mathcal{N}}_{horn}$:	Σ-concept	$\Leftarrow$	Σ-query	$\Leftarrow$	strong Σ-concept	$\Leftrightarrow$	strong Σ-query	$\Leftarrow$	Σ-model

Table 1: Comparing the notions of Σ-entailment in *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$.

and so $\mathcal{T}_1$ does not Σ-concept entail $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{horn}$. ❑

Examples 8 and 11 show that this theorem does not hold for the stronger notions of Σ-entailment. Moreover, for neither *DL-Lite*$^{\mathcal{N}}_{horn}$ nor *DL-Lite*$^{\mathcal{N}}_{bool}$, none of the stronger notions is equivalent to Σ-concept entailment.

The following definition will be used to characterise other Σ-entailments in *DL-Lite*$^{\mathcal{N}}_{horn}$:

Definition 25. A set Ξ of ΣQ-types is said to be *sub-precisely $\mathcal{T}$-realisable* if there is a model $\mathcal{I}$ of $\mathcal{T}$ such that $\mathcal{I}$ realises all the types from Ξ, and every ΣQ-type realised in $\mathcal{I}$ is positively contained in a type from Ξ. We also say that Ξ is *meet-precisely $\mathcal{T}$-realisable* if there is a model $\mathcal{I}$ of $\mathcal{T}$ such that, for every ΣQ-type $\boldsymbol{t}$ realised in $\mathcal{I}$, $\Xi_{\boldsymbol{t}} \neq \emptyset$ and $\boldsymbol{t}^+ = \bigcap_{\boldsymbol{t}_i \in \Xi_{\boldsymbol{t}}} \boldsymbol{t}_i^+$, where $\Xi_{\boldsymbol{t}} = \{\boldsymbol{t}_i \in \Xi \mid \boldsymbol{t}^+ \subseteq \boldsymbol{t}_i^+\}$.

The notion of meet-precise $\mathcal{T}$-realisability is stronger than the notion of sub-precise $\mathcal{T}$-realisability. Indeed, if Ξ is meet-precisely $\mathcal{T}$-realisable then, for each $\boldsymbol{t} \in \Xi$, we have $\Xi_{\boldsymbol{t}} \neq \emptyset$ and so there is $\boldsymbol{t}' \in \Xi_{\boldsymbol{t}}$ with $\boldsymbol{t}^+ \subseteq \boldsymbol{t}'^+$. Therefore Ξ is sub-precisely $\mathcal{T}$-realisable.

Theorem 26. *For any TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$ *and any signature Σ, the following conditions are equivalent*:

(qe$_{\mathbf{h}}$) *$\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$;

(spr) *every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is sub-precisely $\mathcal{T}_2$-realisable.*

Theorem 27. *For any TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$ *and any signature Σ, the following conditions are equivalent*:

(sce$_{\mathbf{h}}$) *$\mathcal{T}_1$ strongly Σ-concept entails $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$;

(sqe$_{\mathbf{h}}$) *$\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{horn}$;

(mpr) *every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is meet-precisely $\mathcal{T}_2$-realisable.*

Example 28. Consider the TBoxes and signature from Example 8. The $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types are $\{\neg\mathsf{Lecturer}\}$ and $\{\mathsf{Lecturer}\}$, and both of them are $\mathcal{T}_2$-realisable. Hence $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{bool}$ (and, therefore, in *DL-Lite*$^{\mathcal{N}}_{horn}$). The singleton $\{\{\mathsf{Lecturer}\}\}$ is precisely $\mathcal{T}_1$-realisable, but not precisely $\mathcal{T}_2$-realisable. Hence $\mathcal{T}_1$ does not Σ-query entail $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{bool}$. However, $\{\{\mathsf{Lecturer}\}\}$ is sub-precisely $\mathcal{T}_1$-realisable and, therefore, $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{horn}$. On the other hand, $\{\{\mathsf{Lecturer}\}\}$ is not meet-precisely $\mathcal{T}_2$-realisable, and so $\mathcal{T}_1$ does not strongly Σ-concept entail $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{horn}$.

Table 1 shows the relative 'strength' of the variants of Σ-entailment introduced in Section 3 for the languages *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$: $\Leftrightarrow$ stands for 'the two notions are equivalent', $\Updownarrow$ for 'the two notions are equivalent for TBoxes formulated in the smaller language', and $\Leftarrow$ and $\Downarrow$ mean that one notion is properly weaker than the other (for TBoxes in the smaller language in case of $\Downarrow$).

4.3. Σ-difference

The semantic criteria formulated above can be used to approximate different variants of Σ-difference between ontologies. When comparing two ontologies with respect to a signature, the ontology engineer needs not only a 'yes' or 'no' answer, but also some informative representation of the difference if the ontologies are different. It is not hard to see that each of the sets $\mathsf{cDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$, $\mathsf{qDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$, $\mathsf{scDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$, $\mathsf{sqDiff}^{\mathcal{L}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ and $\mathsf{mDiff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ of Σ-differences defined in Section 3 is either infinite or empty. Thus, only approximations of these sets can be computed

in practice. One possibility to obtain such approximations is to exploit the semantic criteria provided above. For example, by the criterion of Theorem 20 for Σ-concept difference, the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types that are $\mathcal{T}_1$-realisable but not $\mathcal{T}_2$-realisable are the obvious candidates for inclusion in such an approximation. For each basic $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-concept B, these types contain either B itself or its negation. Of course, as there are exponentially many types in the size of Σ and $\mathcal{T}_1 \cup \mathcal{T}_2$, this method can be unfeasible in practice because there can be too many types to analyse and the resulting list can be incomprehensible. For stronger versions of Σ-difference, one has to consider *sets* of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types; cf. the criteria of Theorems 21, 26 and 27. A detailed investigation of this approach to representing Σ-differences between ontologies is beyond the scope of this paper; we leave it for future research.

5. Robustness properties

The results on Σ-difference and Σ-entailment can be easily misinterpreted if they are not considered in the context of certain robustness properties; moreover, the notions of Σ-difference and Σ-entailment themselves are of limited use if they do not enjoy these properties. In this section, we discuss four types of robustness conditions. First, we consider robustness under definitorial extensions of TBoxes and justify our decision to work with essentially positive existential queries rather than seemingly more natural positive existential queries. Second, we consider preservation results for Σ-entailment under the addition of fresh symbols to Σ and analyse robustness of Σ-inseparability and entailment under taking unions of TBoxes. These two robustness properties are closely related to the interpolation theorem and Robinson's joint consistency property from mathematical logic. Finally, we consider robustness under extensions of the description logic in question with new constructs (which means extensions of the TBox, ABox and query languages). Rather surprisingly, it turns out that in some important cases one can extend the 'lightweight' DL *DL-Lite*$^{\mathcal{N}}_{bool}$ to the very expressive $\mathcal{SHIQ}$ and still preserve Σ-entailment.

5.1. Robustness under definitorial extensions

Recall from Section 2 that in both essentially positive existential queries and ABoxes in *DL-Lite*$^{\mathcal{N}}_{bool}$ we allow *negated* concepts (although negated concepts are not allowed in the case of *DL-Lite*$^{\mathcal{N}}_{horn}$, where we have proper positive existential queries). An alternative approach would be to allow only *positive* concepts (as in *DL-Lite*$^{\mathcal{N}}_{horn}$) or even concept names. As mentioned in Remark 2, these two ways are essentially equivalent in the presence of TBoxes. Yet, they give rise to different notions of Σ-query entailment. Indeed, if only positive concepts are allowed in queries then the TBox $\mathcal{T}_2$ from Example 8 is Σ-query entailed by $\mathcal{T}_1 = \emptyset$, even in *DL-Lite*$^{\mathcal{N}}_{bool}$. We argue, however, that it is the *essentially positive* existential queries that should be considered in the context of this investigation. The reason is that, with only positive queries allowed, the addition of the *definition* $A \equiv \neg\mathsf{Lecturer}$ to both $\mathcal{T}_1$ and $\mathcal{T}_2$ and A to Σ would result in the TBoxes $\mathcal{T}_1'$, $\mathcal{T}_2'$ and signature Σ′ such that $\mathcal{T}_1'$ does not Σ′-query entail $\mathcal{T}_2'$ in *DL-Lite*$^{\mathcal{N}}_{bool}$. This kind of *non-robust* behaviour of Σ-query entailment is clearly undesirable.

To be able to speak about all of our (and perhaps some other) notions of entailment and inseparability at the same time, we introduce the following notation. Given a DL $\mathcal{L}$, we use $\Vdash$ to denote a *ternary entailment relation in* $\mathcal{L}$, whose arguments are two ontologies $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\mathcal{L}$, and a signature Σ. Thus, $\mathcal{T}_1 \Vdash_\Sigma \mathcal{T}_2$ is a shorthand for '$\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ in $\mathcal{L}$'. In other words, $\Vdash$ is the collection of the respective Σ-entailment relations for all signatures Σ. Likewise, we use $\equiv$ to denote a *ternary inseparability relation in* $\mathcal{L}$: $\mathcal{T}_1 \equiv_\Sigma \mathcal{T}_2$ if, and only if, $\mathcal{T}_1 \Vdash_\Sigma \mathcal{T}_2$ and $\mathcal{T}_2 \Vdash_\Sigma \mathcal{T}_1$.

Definition 29. An entailment relation $\Vdash$ in a DL $\mathcal{L}$ is called *robust under definitorial extensions* if, for any signature Σ, $\mathcal{T}_1 \Vdash_\Sigma \mathcal{T}_2$ implies $\mathcal{T}_1 \cup \{A \equiv C\} \Vdash_{\Sigma \cup \{A\}} \mathcal{T}_2 \cup \{A \equiv C\}$ whenever $A \notin sig(\mathcal{T}_1 \cup \mathcal{T}_2)$ and C is a Σ-concept in $\mathcal{L}$.

The proof of the following result is straightforward and left to the reader.

Theorem 30. *All the entailment relations from Section 3 are robust under definitorial extensions in DL-Lite*$^{\mathcal{N}}_{bool}$ *and DL-Lite*$^{\mathcal{N}}_{horn}$.

5.2. Robustness under vocabulary extensions

Clearly, all our entailment relations are preserved under removing symbols from Σ: $\mathcal{T}_1 \Vdash_\Sigma \mathcal{T}_2$ implies $\mathcal{T}_1 \Vdash_{\Sigma'} \mathcal{T}_2$, for any $\Sigma' \subseteq \Sigma$. Obviously, the converse implication does not (and should not) hold in general. However, it turns out that it holds if only *fresh* symbols are added to the signature.

Definition 31. An entailment $\Vdash$ in $\mathcal{L}$ is *robust under vocabulary extensions* if $\mathcal{T}_1 \Vdash_\Sigma \mathcal{T}_2$ implies $\mathcal{T}_1 \Vdash_{\Sigma'} \mathcal{T}_2$, for any Σ and Σ' with $\Sigma' \cap \mathit{sig}(\mathcal{T}_2) \subseteq \Sigma$.

Robustness under vocabulary extensions is of particular importance for Σ-query entailment and the strong versions of Σ-entailment. For example, it means that if $\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ then, for any ABox $\mathcal{A}$, TBox $\mathcal{T}$ and query q containing, apart from symbols in Σ, some *arbitrary* symbols not occurring in $\mathcal{T}_2$, we have $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ whenever $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$. This property is critical for applications, as it is hardly possible to restrict ABoxes and context ontologies to a fixed signature Σ and not permit the use of fresh symbols.

Theorem 32. *All the entailment relations from Section 3 are robust under vocabulary extensions in DL-Lite$^{\mathcal{N}}_{bool}$ and DL-Lite$^{\mathcal{N}}_{horn}$.*

Remark 33. Robustness under vocabulary extensions has important consequences for our investigation of the computational complexity of deciding whether a TBox $\mathcal{T}_1$ Σ-entails another TBox $\mathcal{T}_2$ below. Namely, since $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ if, and only if, $\mathcal{T}_1$ Σ'-entails $\mathcal{T}_2$ for $\Sigma' = \mathit{sig}(\mathcal{T}_2) \cap \Sigma$, we can always assume that $\Sigma \subseteq \mathit{sig}(\mathcal{T}_2)$. Thus, we can take $\ell(\mathcal{T}_1) + \ell(\mathcal{T}_2)$ as the size of the input (and neglect the size of Σ) when measuring the size of the input of the decision problem 'does $\mathcal{T}_1$ Σ-entail $\mathcal{T}_2$?'

Sometimes we will also assume that $\Sigma \subseteq \mathit{sig}(\mathcal{T}_1)$ or even $\Sigma = \mathit{sig}(\mathcal{T}_1)$. The assumption $\Sigma \subseteq \mathit{sig}(\mathcal{T}_1)$ is justified because we can always add $A \sqsubseteq A$ and $\exists P \sqsubseteq \exists P$ to $\mathcal{T}_1$ for all $A, P \in \Sigma$. We can even work with $\Sigma = \mathit{sig}(\mathcal{T}_1)$ because we can uniformly rename all occurrences of concept and role names from $\mathit{sig}(\mathcal{T}_1) \setminus \Sigma$ in $\mathcal{T}_2$ by fresh concept and, respectively, role names, and work with the resulting TBox $\mathcal{T}_2'$ instead of $\mathcal{T}_2$.

5.3. Robustness under joins

Apart from the addition of fresh symbols, it is also important to guarantee robustness under certain joins of ontologies.

Definition 34. An inseparability relation $\equiv$ in $\mathcal{L}$ is *robust under joins* if, for any TBoxes $\mathcal{T}$, $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\mathcal{L}$ and any signature Σ, we have $\mathcal{T} \equiv_\Sigma \mathcal{T}_1 \cup \mathcal{T}_2$ whenever $\mathcal{T} \equiv_\Sigma \mathcal{T}_1$, $\mathcal{T} \equiv_\Sigma \mathcal{T}_2$ and $\mathit{sig}(\mathcal{T}_1) \cap \mathit{sig}(\mathcal{T}_2) \subseteq \Sigma$.

Robustness under joins is of interest for collaborative ontology development. This property means that if two (or more) ontology developers extend an ontology $\mathcal{T}$ independently and do not use common symbols apart from those in a certain signature Σ then they can safely form the union of their extensions $\mathcal{T}_1 \cup \mathcal{T}_2$, provided that they are safe for Σ.

Note that, for robustness under joins, Σ-inseparability between $\mathcal{T}$ and the $\mathcal{T}_i$ is required; as shown in the following example, it is *not* the case that if $\mathcal{T}$ Σ-concept entails $\mathcal{T}_i$, then $\mathcal{T}$ Σ-concept entails $\mathcal{T}_1 \cup \mathcal{T}_2$.

Example 35. Let $\mathcal{T}_1 = \{A \sqsubseteq \exists R, \exists R^- \sqsubseteq B\}$, $\mathcal{T}_2 = \mathcal{T} = \{B \sqsubseteq \bot\}$, and $\Sigma = \{A, B\}$. Then $\mathcal{T}$ Σ-concept entails $\mathcal{T}_i$, $i = 1, 2$, but $\mathcal{T}_1 \cup \mathcal{T}_2 \models A \sqsubseteq \bot$, and so $\mathcal{T}$ does not Σ-concept entail $\mathcal{T}_1 \cup \mathcal{T}_2$.

Theorem 36. *All the inseparability relations from Section 3 are robust under joins in DL-Lite$^{\mathcal{N}}_{bool}$ and DL-Lite$^{\mathcal{N}}_{horn}$.*

Remark 37. Robustness under vocabulary extensions and robustness under joins have been first introduced in [25]. That paper investigates in detail the relationship between these properties and the well-known *Robinson consistency lemma* and *Craig interpolation property* (see, e.g., [45]). For the description logic $\mathcal{EL}$, both robustness conditions are investigated in [44]; and for expressive description logics such as $\mathcal{ALC}$ and its extensions as well first-order logic they are investigated in [25]. Rather interestingly, both robustness properties as well as interpolation typically fail for description logics with nominals and/or role inclusions [46, 25].

5.4. Robustness under language extensions

As we have already seen, in general, language-dependent notions of Σ-entailment do depend on the underlying logic: a stronger logic may induce more differences. So it would be natural to expect that our language-dependent notions of Σ-entailment are not robust under extending *DL-Lite*$^{\mathcal{N}}_{bool}$ to more expressive description logics such as $\mathcal{ALC}$ or $\mathcal{SHIQ}$. Rather surprisingly, it turns out that Σ-query entailment in *DL-Lite*$^{\mathcal{N}}_{bool}$ (and, therefore, strong Σ-query entailment) is robust under extending the *DL-Lite*$^{\mathcal{N}}_{bool}$ language of queries, ABoxes, and context TBoxes to that of $\mathcal{SHIQ}$. In fact, this result will be proved in the appendix for all DLs for which the class of models of TBoxes is closed under *disjoint unions* (see Section A.1 in the appendix for a definition of disjoint unions). We note that typical DLs for which the class of models of TBoxes is not closed under disjoint unions are DLs with nominals and Boolean operators on roles.

We remind the reader that, compared to *DL-Lite*$^{\mathcal{N}}_{bool}$, $\mathcal{SHIQ}$ allows *qualified* number restrictions of the form $\geq q\,R.C$, role inclusion axioms $R_1 \sqsubseteq R_2$, and transitivity constraints stating that certain roles are to be interpreted by transitive relations; see [1] for more details. An ABox in $\mathcal{SHIQ}$ consists of assertions of the form $C(a)$, where C is a $\mathcal{SHIQ}$-concept, and an (essentially positive existential) query in $\mathcal{SHIQ}$ can contain atoms $C(t)$ such that C is a $\mathcal{SHIQ}$-concept. With these auxiliary definitions at hand we can repeat Definition 10 for $\mathcal{L} = \mathcal{SHIQ}$:

Definition 38. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be TBoxes and Σ a signature. We say that $\mathcal{T}_1$ *strongly* Σ*-query entails* $\mathcal{T}_2$ *in* $\mathcal{SHIQ}$ if, for all TBoxes $\mathcal{T}$, ABoxes $\mathcal{A}$ and queries q in $\mathcal{SHIQ}$ with $sig(\mathcal{T} \cup \mathcal{A} \cup \{\mathsf{q}\}) \subseteq \Sigma$ and all tuples $\boldsymbol{a}$ of object names from $\mathcal{A}$, $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ implies $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$.

The following result will be proved in the appendix (Section A.3):

Theorem 39. *For any TBoxes* $\mathcal{T}_1$ *and* $\mathcal{T}_2$ *in DL-Lite*$^{\mathcal{N}}_{bool}$ *and any signature* Σ*, if* $\mathcal{T}_1$ Σ*-query entails* (*or, equivalently, strongly* Σ*-concept entails*) $\mathcal{T}_2$ *in DL-Lite*$^{\mathcal{N}}_{bool}$*, then* $\mathcal{T}_1$ *strongly* Σ*-query entails* $\mathcal{T}_2$ *in* $\mathcal{SHIQ}$*.*

6. Complexity of Σ-entailment

Now we investigate the computational complexity of deciding Σ-entailment (and so Σ-inseparability) between *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$ TBoxes. A first impression of what one can expect is given by Theorem 17 and the known complexity results for deciding Σ-entailment between propositional theories.

6.1. Lower bounds

We remind the reader that the complexity class Π^p_2, also denoted coNP^{NP}, consists of those problems that can be solved by coNP Turing machines with an NP oracle. A typical example of a Π^p_2-complete problem is determining the truth of quantified Boolean formulas (QBFs, for short) of the form $\forall \boldsymbol{p} \exists \boldsymbol{q}\, \varphi(\boldsymbol{p}, \boldsymbol{q})$, where $\varphi(\boldsymbol{p}, \boldsymbol{q})$ is a propositional formula built from propositional variables in the lists $\boldsymbol{p}$ and $\boldsymbol{q}$ (see, e.g., [47, 48]). For example, a propositional formula $\varphi(\boldsymbol{p}, \boldsymbol{q})$ Σ-entails a propositional formula $\psi(\boldsymbol{r}, \boldsymbol{q})$ with disjoint $\boldsymbol{p}$, $\boldsymbol{q}$, $\boldsymbol{r}$ and $\Sigma = \boldsymbol{q}$, if, and only if, the QBF $\forall \boldsymbol{q} \forall \boldsymbol{p} \exists \boldsymbol{r}\, (\varphi(\boldsymbol{p}, \boldsymbol{q}) \rightarrow \psi(\boldsymbol{r}, \boldsymbol{q}))$ is true.

The following theorem is a consequence of the model-theoretic criterion of Example 16; for details and further discussions consult [49].

Theorem 40. *Deciding* Σ*-entailment between propositional theories is* Π^p_2*-complete. Deciding* Σ*-entailment between propositional Horn theories is* coNP*-complete.*

As every propositional theory Φ can trivially be encoded (in linear time) by means of a TBox $\mathcal{T}$ such that Φ and $\mathcal{T}^*$ are logically equivalent (with $\mathcal{T}$ being in *DL-Lite*$^{\mathcal{N}}_{horn}$ whenever Φ is a Horn theory), by Theorems 40 and 17 we obtain the following complexity lower bounds:

Theorem 41. *Deciding* Σ*-entailment in DL-Lite*$^{\mathcal{N}}_{bool}$ *is* Π^p_2*-hard for all of our variants of* Σ*-entailment; deciding* Σ*-entailment in DL-Lite*$^{\mathcal{N}}_{horn}$ *is* coNP*-hard.*

It turns out that these lower bounds actually coincide with the upper bounds for deciding *language-dependent* Σ-entailments (and inseparability) in *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$; we will prove this in Section 6.2. Deciding Σ-model entailment turns out to be a much harder problem. In Section 6.3, we will discuss how to establish decidability of Σ-model entailment in *DL-Lite*$^{\mathcal{N}}_{bool}$ and show that this problem is coNExpTime-hard.

Remark 42. For many expressive DLs as well as $\mathcal{EL}$, the computational complexity of certain notions of Σ-entailment and inseparability is known. Interestingly, even for $\mathcal{EL}$ deciding Σ-entailment is typically much harder than for *DL-Lite*. For example, Σ-concept entailment and Σ-query entailment are both ExpTime-complete for $\mathcal{EL}$ [50, 44]. When moving to more expressive DLs such as $\mathcal{ALC}$ and $\mathcal{ALCQI}$, Σ-query entailment becomes 2ExpTime-complete; and for $\mathcal{ALCQIO}$ Σ-concept entailment becomes undecidable [24, 33].

Σ-model entailment is undecidable for $\mathcal{EL}$ (and all its extensions) [50].

6.2. Complexity of language-dependent Σ-entailments

As we mentioned in Section 2, the satisfiability problem for *DL-Lite*$^{\mathcal{N}}_{bool}$ KBs is NP-complete, while for *DL-Lite*$^{\mathcal{N}}_{horn}$ KBs it is P-complete (under the UNA). It follows that the problem of deciding whether a type $\boldsymbol{t}$ is $\mathcal{T}$-realisable—that is, whether the KB $(\mathcal{T}, \{C(a) \mid C \in \boldsymbol{t}\})$ is satisfiable—is NP-complete for *DL-Lite*$^{\mathcal{N}}_{bool}$ and P-complete for *DL-Lite*$^{\mathcal{N}}_{horn}$. We employ this result and the criterion of Theorem 20 to prove the following:

Theorem 43. *Deciding Σ-concept entailment between DL-Lite*$^{\mathcal{N}}_{bool}$ *TBoxes is* Π^p_2*-complete. Deciding Σ-concept entailment between DL-Lite*$^{\mathcal{N}}_{horn}$ *TBoxes is* coNP*-complete.*

Proof. Let $\mathcal{T}_1$, $\mathcal{T}_2$ be TBoxes in *DL-Lite*$^{\mathcal{N}}_{bool}$ and Σ a signature. By Remark 33, we may assume without loss of generality that $\Sigma \subseteq sig(\mathcal{T}_1 \cup \mathcal{T}_2)$. By Theorem 20, the following algorithm decides whether $\mathcal{T}_1$ *does not* Σ-concept entail $\mathcal{T}_2$:

1. Guess a $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type $\boldsymbol{t}$. (Observe that the size of $\boldsymbol{t}$ is linear in the size of $\mathcal{T}_1 \cup \mathcal{T}_2$.)
2. Check, by calling an NP-oracle, whether (i) $\boldsymbol{t}$ is $\mathcal{T}_1$-realisable and whether (ii) $\boldsymbol{t}$ is not $\mathcal{T}_2$-realisable.
3. Return '$\mathcal{T}_1$ does not Σ-concept entails $\mathcal{T}_2$' if the answers to (i) and (ii) are both positive.

It should be clear that this algorithm runs in Σ^p_2, and so the problem of deciding whether $\mathcal{T}_1$ *does* Σ-concept entail $\mathcal{T}_2$ is in Π^p_2.

The same algorithm, calling a P-oracle for *DL-Lite*$^{\mathcal{N}}_{horn}$ TBoxes, runs in NP and so the problem of deciding, given *DL-Lite*$^{\mathcal{N}}_{horn}$ TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$, whether $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ is in coNP. ❑

To check the criterion of Theorem 21 for the other language-dependent variants of Σ-entailment, we should be able to establish precise realisability of sets of types. The following simple example illustrates the intuition behind the notions we need to do this.

Example 44. Let $\Sigma = \{A, B\}$, $Q = \{1\}$, $\mathcal{T} = \{A \equiv \exists P,\ \ \exists P^- \sqsubseteq B,\ \ \exists R^- \sqsubseteq B,\ \ B \sqsubseteq \exists R\}$, and suppose that we want to know whether the set $\Xi = \{\boldsymbol{t}'_0, \boldsymbol{t}'_1\}$ of the ΣQ-types $\boldsymbol{t}'_0 = \{A, \neg B\}$ and $\boldsymbol{t}'_1 = \{\neg A, B\}$ is $\mathcal{T}$-realisable. In other words, we would like to know whether there is a model $\mathcal{I}$ of $\mathcal{T}$ with points x_0 and x_1 such that $\boldsymbol{t}'_i \subseteq \boldsymbol{t}_i = \boldsymbol{t}_{\mathcal{I}}(x_i)$, $i = 0, 1$, where $\boldsymbol{t}_{\mathcal{I}}(x_i)$ is the $sig(\mathcal{T})Q$-type of x_i defined by (1). If such $\mathcal{I}$ and the x_i do exist then, clearly, $\exists P \in \boldsymbol{t}_0$ and $\exists R \in \boldsymbol{t}_1$, which means that a P-arrow starts from x_0 and an R-arrow starts from x_1. But then there must exist 'witness types' for the ends of these arrows, that is, some $sig(\mathcal{T})Q$-types $\boldsymbol{t}_{\exists P^-}$ and $\boldsymbol{t}_{\exists R^-}$ containing $\exists P^-$ and $\exists R^-$, respectively. By the axioms of $\mathcal{T}$, both of these types must contain B and $\exists R$, which again requires a witness type for $\exists R^-$, e.g., the same $\boldsymbol{t}_{\exists R^-}$. In fact, the types $\boldsymbol{t}_0$, $\boldsymbol{t}_1$, $\boldsymbol{t}_{\exists P^-}$ and $\boldsymbol{t}_{\exists R^-}$ are all we need to construct an infinite forest-like model of $\mathcal{T}$ realising Ξ and precisely realising the set $\{\boldsymbol{t}_0, \boldsymbol{t}_1, \boldsymbol{t}_{\exists P^-}, \boldsymbol{t}_{\exists R^-}\}$. This construction known as 'unravelling' (of the appropriate part of $\mathcal{I}$) is shown in Fig. 2.

It is not hard to see that in general, for a TBox with m role names, this unravelling procedure, having started with k types, will produce a model with at most $k + 2m$ distinct types. More precisely, a set Ξ' of ΣQ-types is $\mathcal{T}$-realisable (with $\Sigma \subseteq sig(\mathcal{T})$ and $Q_{\mathcal{T}} \subseteq Q$) if, and only if, there is a precisely $\mathcal{T}$-realisable set Ξ of $sig(\mathcal{T})Q$-types such that (i) $|\Xi| \leq |\Xi'| + 2m$, where m is the number of role names in $\mathcal{T}$, and (ii) each type in Ξ' can be extended to a type in Ξ.

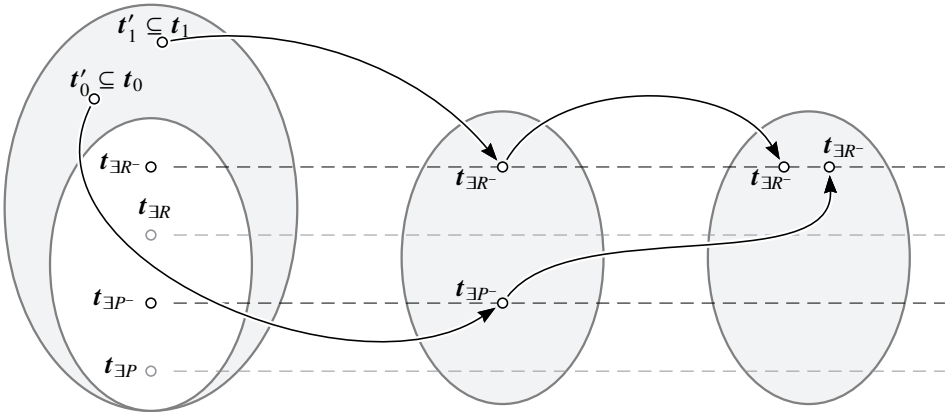

Figure 2: The first three steps of constructing a model realising $\{t'_0, t'_1\}$ by means of unravelling.

This example motivates the following definition. Let $\mathcal{T}$ be a TBox in *DL-Lite*$^{\mathcal{N}}_{bool}$, Σ a signature and Q a set of natural numbers with $\Sigma \subseteq sig(\mathcal{T})$ and $Q_{\mathcal{T}} \subseteq Q$.

Definition 45. Given a set $\Xi = \{t'_0, \dots, t'_k\}$ of $(k+1)$ ΣQ-types, a pair of sequences $\Xi^{\mathcal{T}} = ((t_0, \dots, t_k), (t_{k+1}, \dots, t_{k+2m}))$ of (not necessarily distinct) $sig(\mathcal{T})Q$-types is called a *$\mathcal{T}$-witness for* Ξ if m is the number of role names in $\mathcal{T}$ and the following conditions are satisfied:

($\mathbf{w}_1$) $t'_i = t_i \restriction_\Sigma$, for $0 \le i \le k$;

($\mathbf{w}_2$) each type in the sequence $t_0, \dots, t_k, t_{k+1}, \dots, t_{k+2m}$ is $\mathcal{T}$-realisable;

($\mathbf{w}_3$) for each role name P_i in $\mathcal{T}$ $(1 \le i \le m)$, the types t_{k+2i-1} and t_{k+2i} are witnesses for $\exists P_i$ and $\exists P_i^-$, respectively; more precisely,

$$\exists P_i \in t_{k+2i-1} \text{ and } \exists P_i^- \in t_{k+2i} \quad \text{whenever} \quad \{\exists P_i^-, \exists P_i\} \cap t_j \neq \emptyset, \text{ for some } 0 \le j \le k + 2m.$$

A $\mathcal{T}$-witness $\Xi^{\mathcal{T}}$ is called a *precise $\mathcal{T}$-witness* for Ξ if

(w-pr) for every type t_i in the sequence $t_{k+1}, \dots, t_{k+2m}$, there is a type $t'_j \in \Xi$ such that $t_i \restriction_\Sigma = t'_j$.

It follows from the definition and the unravelling construction of Example 44 (see also [9]) that we have:

Proposition 46. (i) *A set Ξ of ΣQ-types is $\mathcal{T}$-realisable if, and only if, there is a $\mathcal{T}$-witness for Ξ.*
(ii) *A set Ξ of ΣQ-types is precisely $\mathcal{T}$-realisable if, and only if, there is a precise $\mathcal{T}$-witness for Ξ.*

Recall now that, by Theorem 21, to check whether $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$, we have to verify the condition:

(pr) every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is precisely $\mathcal{T}_2$-realisable.

The unravelling construction illustrated in Example 44 and Proposition 46 indicate, however, that instead of considering arbitrary $\mathcal{T}_1$-realisable *sets* of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types, we can deal with $\mathcal{T}_1$-witnesses generated by a single $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type only. More precisely, we can simplify **(pr)** to the following criterion:

Theorem 47. *$\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in DL-Lite*$^{\mathcal{N}}_{bool}$ *if, and only if, the following condition holds, where m_1 is the number of role names in $\mathcal{T}_1$:*

($\mathbf{pr}'$) *for every $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type t, if there is a $\mathcal{T}_1$-witness $((t_0), (t_1, \dots, t_{2m_1}))$ for $\{t\}$, then there is a precise $\mathcal{T}_2$-witness for the set $\{t_0 \restriction_\Sigma, t_1 \restriction_\Sigma, \dots, t_{2m_1} \restriction_\Sigma\}$.*

Proof. **(pr)** $\Rightarrow$ **(pr′)**. If $((\boldsymbol{t}_0),(\boldsymbol{t}_1,\dots,\boldsymbol{t}_{2m_1}))$ is a $\mathcal{T}_1$-witness for $\{\boldsymbol{t}\}$ then, clearly, the set $\{\boldsymbol{t}_0\restriction_\Sigma,\boldsymbol{t}_1\restriction_\Sigma,\dots,\boldsymbol{t}_{2m_1}\restriction_\Sigma\}$ is precisely $\mathcal{T}_1$-realisable. By **(pr)**, this set is precisely $\mathcal{T}_2$-realisable, and so, by Proposition 46, it has a precise $\mathcal{T}_2$-witness.

(pr′) $\Rightarrow$ **(pr)**. Suppose now that a set $\Xi=\{\boldsymbol{t}'_0,\dots,\boldsymbol{t}'_k\}$ of $\Sigma Q_{\mathcal{T}_1\cup\mathcal{T}_2}$-types is precisely $\mathcal{T}_1$-realisable. By Proposition 46, there exists a precise $\mathcal{T}_1$-witness $\Xi^{\mathcal{T}_1}=((\boldsymbol{t}_0,\dots,\boldsymbol{t}_k),(\boldsymbol{t}_{k+1},\dots,\boldsymbol{t}_{k+2m_1}))$ for Ξ. Then, clearly, the sequences $((\boldsymbol{t}_i),(\boldsymbol{t}_{k+1},\dots,\boldsymbol{t}_{k+2m_1}))$ are $\mathcal{T}_1$-witnesses for the singletons $\{\boldsymbol{t}_i\}$, $0\le i\le k$. According to **(pr′)** and Proposition 46, there are models $\mathcal{J}_i$ of $\mathcal{T}_2$ precisely realising the sets $\Xi_i=\{\boldsymbol{t}_i\restriction_\Sigma,\boldsymbol{t}_{k+1}\restriction_\Sigma,\dots,\boldsymbol{t}_{k+2m_1}\restriction_\Sigma\}\subseteq\Xi$. Now, define an interpretation $\mathcal{I}$ as the *disjoint union* of the $\mathcal{J}_i$ (a precise definition is given in the appendix, Section A.1). As $\bigcup_{i=0}^{k}\Xi_i=\Xi$, it is easy to see that $\mathcal{I}$ is a model of $\mathcal{T}_2$ precisely realising Ξ. ❑

Note that the size of witnesses mentioned in condition **(pr′)** is linear in the size of $\mathcal{T}_1$ and $\mathcal{T}_2$: the size of the $\mathcal{T}_1$-witness is $1+2m_1$ and the size of the $\mathcal{T}_2$-witness is $1+2m_1+2m_2$, where m_i is the number of roles names in $\mathcal{T}_i$, $i=1,2$. We will use this observation to construct a Π_2^p algorithm deciding Σ-query and the strong forms of entailment between *DL-Lite*$_{bool}^{\mathcal{N}}$ TBoxes. But before that let us see how to modify the notions of witnesses for *DL-Lite*$_{horn}^{\mathcal{N}}$ TBoxes.

Definition 48. Given a TBox $\mathcal{T}$ in *DL-Lite*$_{horn}^{\mathcal{N}}$ and a set Ξ of ΣQ-types, by a *sub-precise $\mathcal{T}$-witness* for Ξ we understand any $\mathcal{T}$-witness $\Xi^{\mathcal{T}}=((\boldsymbol{t}_0,\dots,\boldsymbol{t}_k),(\boldsymbol{t}_{k+1},\dots,\boldsymbol{t}_{k+2m}))$ for Ξ satisfying the following condition:

(w-spr) for every type $\boldsymbol{t}_i$ in the sequence $\boldsymbol{t}_{k+1},\dots,\boldsymbol{t}_{k+2m}$, there is a type $\boldsymbol{t}'_j\in\Xi$ such that $\boldsymbol{t}_i^+\restriction_\Sigma\subseteq(\boldsymbol{t}'_j)^+$.

A *meet-precise $\mathcal{T}$-witness* for Ξ is any $\mathcal{T}$-witness $\Xi^{\mathcal{T}}=((\boldsymbol{t}_0,\dots,\boldsymbol{t}_k),(\boldsymbol{t}_{k+1},\dots,\boldsymbol{t}_{k+2m}))$ for Ξ such that

(w-mpr) for every type $\boldsymbol{t}_i$ in $\boldsymbol{t}_{k+1},\dots,\boldsymbol{t}_{k+2m}$, the set $\Xi_{\boldsymbol{t}_i}=\{\boldsymbol{t}'_j\in\Xi\mid\boldsymbol{t}_i^+\restriction_\Sigma\subseteq(\boldsymbol{t}'_j)^+\}$ is nonempty and $\boldsymbol{t}_i^+\restriction_\Sigma=\bigcap_{\boldsymbol{t}'_j\in\Xi_{\boldsymbol{t}_i}}(\boldsymbol{t}'_j)^+$.

It follows from the definition and the unravelling construction that we have:

Proposition 49. (i) *A set Ξ of ΣQ-types is sub-precisely $\mathcal{T}$-realisable if, and only if, there is a sub-precise $\mathcal{T}$-witness for Ξ.*

(ii) *A set Ξ of ΣQ-types is meet-precisely $\mathcal{T}$-realisable if, and only if, there is a meet-precise $\mathcal{T}$-witness for Ξ.*

Since realisability of a single type with respect to a TBox in *DL-Lite*$_{horn}^{\mathcal{N}}$ can be checked in deterministic polynomial time and the number of types in witnesses is linear in the number of roles (and thus, in the length of the TBox), one can check whether a given set has a precise (or sub- or meet-precise) witness and compute it in polynomial time (a proof can be found in the technical appendix, Section A.4):

Lemma 50. *There is an algorithm which, given a TBox $\mathcal{T}$ in DL-Lite*$_{horn}^{\mathcal{N}}$ *and a set Ξ of ΣQ-types with $Q\supseteq Q_{\mathcal{T}}$, decides in deterministic polynomial time whether Ξ has a precise, sub-precise or meet-precise $\mathcal{T}$-witness and constructs such a witness if it exists.*

Similarly to Theorem 47, conditions **(spr)** and **(mpr)** of Theorems 26 and 27 can be equivalently reformulated in terms of sub- and meet-precise witnesses:

Theorem 51. (i) *$\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in DL-Lite*$_{horn}^{\mathcal{N}}$ *if, and only if, the following condition holds, where m_1 is the number of role names in $\mathcal{T}_1$:*

(spr′) *for every $\Sigma Q_{\mathcal{T}_1\cup\mathcal{T}_2}$-type $\boldsymbol{t}$, if there is a $\mathcal{T}_1$-witness $((\boldsymbol{t}_0),(\boldsymbol{t}_1,\dots,\boldsymbol{t}_{2m_1}))$ for $\{\boldsymbol{t}\}$ then there is a sub-precise $\mathcal{T}_2$-witness for the set $\{\boldsymbol{t}_0\restriction_\Sigma,\boldsymbol{t}_1\restriction_\Sigma,\dots,\boldsymbol{t}_{2m_1}\restriction_\Sigma\}$;*

(ii) *$\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ in DL-Lite*$_{horn}^{\mathcal{N}}$ *if, and only if, the following condition holds*:

(mpr′) *for every $\Sigma Q_{\mathcal{T}_1\cup\mathcal{T}_2}$-type $\boldsymbol{t}$, if there is a $\mathcal{T}_1$-witness $((\boldsymbol{t}_0),(\boldsymbol{t}_1,\dots,\boldsymbol{t}_{2m_1}))$ for $\{\boldsymbol{t}\}$ then there is a meet-precise $\mathcal{T}_2$-witness for the set $\{\boldsymbol{t}_0\restriction_\Sigma,\boldsymbol{t}_1\restriction_\Sigma,\dots,\boldsymbol{t}_{2m_1}\restriction_\Sigma\}$.*

We are now in a position to obtain the following tight complexity results:

Theorem 52. (i) *Deciding* Σ*-query* (*and so strong* Σ*-concept and strong* Σ*-query*) *entailment between DL-Lite*$^{\mathcal{N}}_{bool}$ *TBoxes is* Π^p_2*-complete.*

(ii) *Deciding* Σ*-query, strong* Σ*-concept and strong* Σ*-query entailments between DL-Lite*$^{\mathcal{N}}_{horn}$ *TBoxes is* coNP*-complete.*

Proof. (i) Let $\mathcal{T}_1$, $\mathcal{T}_2$ be TBoxes in *DL-Lite*$^{\mathcal{N}}_{bool}$ and Σ a signature. By Remark 33, we may assume without loss of generality that $\Sigma \subseteq sig(\mathcal{T}_1 \cup \mathcal{T}_2)$. By Theorem 47, the following algorithm decides whether $\mathcal{T}_1$ *does not* Σ-query entail $\mathcal{T}_2$:

1. Guess a $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type $\boldsymbol{t}$. (Observe that the size of types is linear in the size of $\mathcal{T}_1 \cup \mathcal{T}_2$.)
2. Check, by calling an NP-oracle, whether (a) there is a precise $\mathcal{T}_1$-witness $((\boldsymbol{t}_0), (\boldsymbol{t}_1, \dots, \boldsymbol{t}_{2m_1}))$ for $\{\boldsymbol{t}\}$ and (b) there is *no* precise $\mathcal{T}_2$-witness for $\{\boldsymbol{t}_0 \restriction_\Sigma, \boldsymbol{t}_1 \restriction_\Sigma, \dots, \boldsymbol{t}_{2m_1} \restriction_\Sigma\}$.
3. Return '$\mathcal{T}_1$ does not Σ-query entails $\mathcal{T}_2$' if the answers to (a) and (b) are both positive.

This algorithm runs in Σ^p_2, and so the problem of deciding whether $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ is in Π^p_2.

(ii) Here we use a similar algorithm, calling a P-oracle of Lemma 50 to compute for a given set of types a sub-precise or meet-precise witness. This algorithm clearly runs in NP. ❑

6.3. Decidability of Σ*-model entailment*

In this section, we show that Σ-model entailment between TBoxes in *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$ is decidable, but coNExpTime-hard. The decidability proof is by embedding in the two-sorted first-order theory of Boolean algebras (BA) combined with Presburger arithmetic (PA) for representing cardinalities of sets. The decidability of this theory, called BAPA, has been first proved in [51]. The computational complexity and practical algorithms for BAPA have been investigated in [52]. The coNExpTime lower bound is proved by a reduction of the model conservativity problem for the modal logic $\mathcal{S}5$, which is known to be coNExpTime-complete [53]. As reasoning in BAPA is known to be harder than coNExpTime and our encoding is exponential in the worst case, the precise computational complexity of Σ-model entailment remains open. We will provide a brief discussion of the feasibility of using Σ-model entailment in practice at the end of this section.

Let us begin by expanding Example 14 and showing that *uncountable* models have to be considered when deciding Σ-model entailment.

Example 53. Let $\mathcal{T}_1$ be a *DL-Lite*$^{\mathcal{N}}_{horn}$ TBox stating, using auxiliary role names R and R_B, that a concept B is infinite:

$$\mathcal{T}_1 = \{\top \sqsubseteq \exists R,\ \exists R^- \sqsubseteq \exists R_B,\ \exists R^- \sqcap \exists R_B^- \sqsubseteq \bot,\ \exists R_B^- \sqsubseteq B,\ B \sqsubseteq \exists R_B,\ \geq 2\, R_B^- \sqsubseteq \bot\}.$$

For $\mathcal{T}_2$ we take the same TBox as in Example 14 stating that P is an injection from A to B:

$$\mathcal{T}_2 = \{A \equiv \exists P,\ \ \exists P^- \sqsubseteq B,\ \ \geq 2\, P \sqsubseteq \bot,\ \ \geq 2\, P^- \sqsubseteq \bot\}.$$

Let $\Sigma = \{A, B\}$. There exists an uncountable model $\mathcal{I}$ of $\mathcal{T}_1$ with uncountable $A^{\mathcal{I}}$ and countable $B^{\mathcal{I}}$. Thus, there is no injection from $A^{\mathcal{I}}$ to $B^{\mathcal{I}}$, and so $\mathcal{I} \restriction_\Sigma \in \mathsf{mDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ and $\mathcal{T}_1$ does not Σ-model entail $\mathcal{T}_2$. It is of interest to observe, however, that if $\mathcal{I}$ is an at most countably infinite model of $\mathcal{T}_1$, then there is always an injection from $A^{\mathcal{I}}$ to $B^{\mathcal{I}}$. Thus, in this case there exists a model $\mathcal{I}'$ of $\mathcal{T}_2$ that coincides with $\mathcal{I}$ on Σ. Using our semantic criteria, it is readily checked that $\mathcal{T}_1$ Σ-entails $\mathcal{T}_2$ for any of the language-dependent notions of Σ-entailment.

This example shows that to decide Σ-model entailment we have to reason effectively about (possibly infinite) cardinalities of sets and, of course, basic set-theoretic operations such as intersection and complement. BAPA is a two-sorted first-order language designed precisely for this purpose.

Formally, the language of BAPA is defined as follows. Its *terms of sort* set are constructed from variables $X_1, X_2, \dots$ and constants $\mathbf{0}$ (the empty set) and $\mathbf{1}$ (the whole set) using the binary function symbols $\cap$ (intersection), $\cup$ (union), and the unary function symbol $\bar{\cdot}$ (complement). The *terms of sort* number are constructed from variables $x_1, x_2, \dots$, constants from the set $K = \{0, 1, 2, \dots\}$ for natural numbers, and expressions $|B|$, for B a term of sort set, using the binary function symbol $+$. As usual, we prefer the infix notation for the binary function symbols and write, e.g., $X \cap Y$ instead of $\cap(X, Y)$. *Atomic BAPA formulas* are of the form:

– $B_1 = B_2$ and $B_1 \subseteq B_2$, where B_1 and B_2 are terms of sort set;

– $|B| = k$ and $|B| \geq k$, where B is a term of sort set and k a term of sort number;

– $k_1 \leq k_2$, where k_1 and k_2 are of sort number.

BAPA formulas are now constructed in the standard way using first-order quantification (for variables of sort set and number), conjunction and negation.

We are interested in the validity of BAPA formulas in two-sorted relational structures, called *BAPA structures*, of the form

$$\mathfrak{A} = ((2^\Delta, \cap, \cup, \bar{\cdot}, \emptyset, \Delta), (\mathsf{card}(\Delta), +, 0, 1, 2, \dots), |\cdot|),$$

where $(2^\Delta, \cap, \cup, \bar{\cdot}, \emptyset, \Delta)$ is the Boolean algebra of subsets of a nonempty set Δ, $\mathsf{card}(\Delta)$ is the set of cardinal numbers $\{\kappa \mid \kappa \leq |\Delta|\}$, and $|B|$ is the cardinality of a subset B of Δ. A *BAPA model* $\mathfrak{M}$ consists of a BAPA structure $\mathfrak{A}$, an *interpretation* $X_i^\mathfrak{M} \subseteq \Delta$ of the variables X_i of sort set as subsets of Δ, and an *interpretation* $x_i^\mathfrak{M} \in \{n \mid n \leq |\Delta|\}$ of the variables x_i of sort number as cardinal numbers that are not greater than the cardinality of Δ. (The overloading of symbols here is deliberate.)

Decidability of validity of BAPA formulas follows from [51]:

Theorem 54. *The problem whether a BAPA sentence is true in all BAPA models is decidable.*

In fact, it follows from [52] that the validity problem for BAPA sentences is in 2ExpSpace. We now give a reduction of Σ-model entailment in $\mathcal{DL}\text{-}\mathit{Lite}_{bool}^{\mathcal{N}}$ to validity of BAPA sentences. As BAPA does not have binary relation symbols, the main problem is to encode the truth conditions for number restrictions $\geq q\, R$ as BAPA sentences.

Suppose that TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\mathcal{DL}\text{-}\mathit{Lite}_{bool}^{\mathcal{N}}$ and a signature Σ are given. By Remark 33, we may assume without loss of generality that $\Sigma = \mathit{sig}(\mathcal{T}_1)$.

For every basic concept B occurring in $\mathcal{T}_1 \cup \mathcal{T}_2$, we take a BAPA variable X_B of sort set and then, for every concept C in the signature of $\mathcal{T}_1 \cup \mathcal{T}_2$, define inductively a BAPA term C^s of sort set:

$$B^s = X_B, \qquad \bot^s = \mathbf{0}, \qquad \top^s = \mathbf{1},$$
$$(\neg C)^s = (\overline{C})^s, \qquad (C_1 \sqcap C_2)^s = C_1^s \cap C_2^s.$$

We also set, for $i = 1, 2$,

$$\mathcal{T}_i^s = \{C_1^s \subseteq C_2^s \mid C_1 \sqsubseteq C_2 \in \mathcal{T}_i\}.$$

As a first approximation, we can try and translate the problem whether $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$ as the validity problem for the BAPA sentences of the form

$$\forall \boldsymbol{X} \Big(\bigwedge_{\alpha \in \mathcal{T}_1^s} \alpha \;\; \rightarrow \;\; \exists \boldsymbol{Y} \bigwedge_{\alpha \in \mathcal{T}_2^s} \alpha \Big), \tag{2}$$

where $\boldsymbol{X}$ is the sequence of variables of sort set occurring in $\mathcal{T}_1^s$ and $\boldsymbol{Y}$ is the sequence of variables of sort set that occur in $\mathcal{T}_2^s$ but not in $\mathcal{T}_1^s$. This sentence is supposed to convey the meaning of '$\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$' for $\Sigma = \mathit{sig}(\mathcal{T}_1)$: every Σ-model of $\mathcal{T}_1$ can be extended to a model of $\mathcal{T}_2$ (cf. Definition 13). The problem, however, is that our encoding does not take into account the semantics of number restrictions.

Let $q_{\max}$ be the maximal numerical parameter occurring in $\mathcal{T}_1 \cup \mathcal{T}_2$; if there are no such parameters then we set $q_{\max} = 0$. For every role name P in $\mathcal{T}_1 \cup \mathcal{T}_2$, we introduce two sets of additional fresh variables of sort set: for $R = P$ and $R = P^-$, set

$$\mathcal{X}_R = \{X_{=q\,R} \mid 0 \leq q \leq q_{\max}\} \cup \{X_{>q_{\max}\,R}\}.$$

Intuitively, we want $X_{=q\,R}$ to stand for the set of points with precisely q R-successors, and $X_{>q_{\max}\,R}$ for the set of points with more than $q_{\max}$ R-successors. To ensure this, we first add to $\mathcal{T}_i^s$ the following (obviously sound) equations, for every role name P in $\mathcal{T}_i$ and $R \in \{P, P^-\}$:

– $X \cap X' = \mathbf{0}$, for any two distinct $X, X' \in \mathcal{X}_R$;

– $X_{=0\,R} \cup \dots \cup X_{=q_{\max}\,R} \cup X_{>q_{\max}\,R} = \mathbf{1}$;

– $X_{\geq q\,R} = X_{=q\,R} \cup X_{=(q+1)\,R} \cup \dots \cup X_{=q_{\max}\,R} \cup X_{>q_{\max}\,R}$, for every $\geq q\,R$ in $\mathcal{T}_i$.

The resulting sets will be still denoted by $\mathcal{T}_i^s$. It remains to formulate relationships between the cardinality of the interpretations of variables in $\mathcal{X}_P$ and $\mathcal{X}_{P^-}$. For a binary relation ϱ, define the *ϱ-outdegree* $o_\varrho(d)$ of a point d as $o_\varrho(d) = |\{d' \mid (d, d') \in \varrho\}|$; the *$\varrho$-indegree* of d is $i_\varrho(d) = |\{d' \mid (d', d) \in \varrho\}|$.

Definition 55. A *set-system*

$$\mathcal{S} \;=\; (A_1, \dots, A_{q_{\max}}, A_\infty), (B_1, \dots, B_{q_{\max}}, B_\infty)$$

for $\mathcal{T}_1$ and $\mathcal{T}_2$ consists of two finite sequences of sets such that the sets in each sequence are mutually disjoint. A binary relation ϱ is called a *solution* to $\mathcal{S}$ if

– A_q is the set of points of ϱ-outdegree q, for $1 \leq q \leq q_{\max}$, A_∞ is the set of points of ϱ-outdegree $> q_{\max}$;

– B_q is the set of points of ϱ-indegree q, for $1 \leq q \leq q_{\max}$, B_∞ is the set of points of ϱ-indegree $> q_{\max}$.

The following result will be proved in the appendix, Section A.4.

Lemma 56. *For every role name P and every number $q_{\max} \geq 1$, one can construct a BAPA formula $\varphi_{P,q_{\max}}$ with free variables*

$$X_{=1\,P}, \dots, X_{=q_{\max}\,P}, X_{>q_{\max}\,P},\; X_{=1\,P^-}, \dots, X_{=q_{\max}\,P^-}, X_{>q_{\max}\,P^-}$$

such that, for every BAPA model $\mathfrak{M}$, the following conditions are equivalent:

(i) $\mathfrak{M} \models \varphi_{P,q_{\max}}$;

(ii) *the set-system* $(X^{\mathfrak{M}}_{=1\,P}, \dots, X^{\mathfrak{M}}_{=q_{\max}\,P}, X^{\mathfrak{M}}_{>q_{\max}\,P}), (X^{\mathfrak{M}}_{=1\,P^-}, \dots, X^{\mathfrak{M}}_{=q_{\max}\,P^-}, X^{\mathfrak{M}}_{>q_{\max}\,P^-})$ *has a solution.*

Given this lemma, we can rectify (2) in a straightforward way. Let $\boldsymbol{X}$ be the sequence of variables of sort set occurring in $\mathcal{T}_1^s$ or $\mathcal{X}_P \cup \mathcal{X}_{P^-}$, for P in $\mathcal{T}_1$, and let $\boldsymbol{Y}$ be the sequence of variables of sort set occurring in $\mathcal{T}_2^s$ or $\mathcal{X}_P \cup \mathcal{X}_{P^-}$, for P in $\mathcal{T}_2$, but not in $\boldsymbol{X}$. Then we define a BAPA formula $\varphi_{\mathcal{T}_1,\mathcal{T}_2}$ by taking

$$\varphi_{\mathcal{T}_1,\mathcal{T}_2} \;=\; \forall \boldsymbol{X}\, \Big(\big(\bigwedge_{\alpha \in \mathcal{T}_1^s} \alpha \wedge \bigwedge_{P \in sig(\mathcal{T}_1)} \varphi_{P,q_{\max}} \big) \;\rightarrow\; \exists \boldsymbol{Y}\, \big(\bigwedge_{\alpha \in \mathcal{T}_2^s} \alpha \wedge \bigwedge_{P \in sig(\mathcal{T}_2)} \varphi_{P,q_{\max}} \big) \Big),$$

where $q_{\max}$ is the maximal numerical parameter occurring in $\mathcal{T}_1 \cup \mathcal{T}_2$ (if $q_{\max} = 0$ then $\varphi_{\mathcal{T}_1,\mathcal{T}_2}$ has the form (2)). Now, it is straightforward to prove the following result:

Theorem 57. *Let $\Sigma = sig(\mathcal{T}_1)$. Then $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$ if, and only if, $\varphi_{\mathcal{T}_1,\mathcal{T}_2}$ is valid.*

It follows from Remark 33 and the decidability of BAPA that Σ-model entailment is decidable. The formula $\varphi_{P,q_{\max}}$ constructed in the proof of Lemma 56 is exponential in the size of $q_{\max}$, and so the upper bound for the computational complexity of deciding Σ-model entailment is 'disappointing' 3ExpSpace. In the appendix (Section A.4) we establish, using a reduction of the model conservativity problem for modal logic $\mathcal{S}5$, the following lower bound:

Theorem 58. *Deciding Σ-model entailment for TBoxes in DL-Lite$^{\mathcal{N}}_{horn}$ with maximal numerical parameter $q_{\max} = 3$ is* coNExpTime*-hard.*

Finding tight complexity bounds for deciding Σ-model entailment remains an open problem that is beyond the scope of this paper. As the encoding of Σ-model entailment into BAPA uses only very little arithmetic (exhibited in the construction of $\varphi_{P,q_{\max}}$ in the proof of Lemma 56), we conjecture that the complexity is actually between coNExpTime and ExpSpace. It is important to note that the formula $\varphi_{P,q_{\max}}$ constructed in Lemma 56 is of polynomial size if the maximal parameter $q_{\max}$ is fixed. This appears to be a natural assumption, as in many cases number restrictions are only used to introduce functional roles. It would be of interest to conduct experiments for such TBoxes and Σ-model entailment using the encoding into BAPA above and, say, the BAPA reasoner introduced in [52].

7. Inseparability modules

In this section, we discuss how the notions of Σ-inseparability can be employed to define modules, analyse relationships between modules, and design module extraction algorithms. Intuitively, a module of a TBox $\mathcal{T}$ is a subset $\mathcal{M}$ of $\mathcal{T}$ that says the same about a certain subject matter as the whole $\mathcal{T}$. Assuming that subject matters are represented by signatures Σ and that 'saying the same about Σ' is formalised as being Σ-inseparable, we come to modules that are Σ-inseparable from the TBoxes containing them.

Many interesting properties of such modules and even module extraction algorithms can be described without referring to a particular notion of Σ-inseparability, but only using certain properties of inseparability relations. So, like in Section 5, we will consider some abstract notion ≡ of inseparability relation in a DL $\mathcal{L}$, which covers all the variants of Σ-inseparability introduced above, and develop the corresponding notions of modules within this framework. One obvious property of ≡ we need is that it is an *equivalence relation*. As before, we assume that $\mathcal{L}$ is one of the logics *DL-Lite*$^{\mathcal{N}}_{bool}$ or *DL-Lite*$^{\mathcal{N}}_{horn}$.

Definition 59. We call an inseparability relation ≡ in $\mathcal{L}$ *monotone* if it satisfies the following two conditions, for all TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\mathcal{L}$ and all signatures Σ:

(M_{sig}) for any $\Sigma' \subseteq \Sigma$, if $\mathcal{T}_1 \equiv_\Sigma \mathcal{T}_2$ then $\mathcal{T}_1 \equiv_{\Sigma'} \mathcal{T}_2$;

($M_\top$) for any TBox $\mathcal{T}$ in $\mathcal{L}$, if $\mathcal{T}_1 \subseteq \mathcal{T} \subseteq \mathcal{T}_2$ and $\mathcal{T}_1 \equiv_\Sigma \mathcal{T}_2$, then $\mathcal{T} \equiv_\Sigma \mathcal{T}_1$.

Condition (M_{sig}) formalises the intuition that if two TBoxes are Σ-inseparable then they are Σ′-inseparable for any smaller signature Σ′; ($M_\top$) demands that any TBox sandwiched between two inseparable TBoxes should be inseparable from either of them. The following statement is left to the reader as an easy exercise.

Theorem 60. *All the inseparability relations in DL-Lite*$^{\mathcal{N}}_{bool}$ *and DL-Lite*$^{\mathcal{N}}_{horn}$ *from Section 3 are monotone.*

We now introduce and discuss three notions of modules induced by an inseparability relation. The first one formalises the intuition discussed above, whereas the other two take into account some additional properties one might want modules to have.

Definition 61. Let ≡ be an inseparability relation in $\mathcal{L}$, $\mathcal{T}$ a TBox in $\mathcal{L}$, $\mathcal{M} \subseteq \mathcal{T}$, and Σ a signature. We say that $\mathcal{M}$ is

- a $\equiv_\Sigma$-*module of* $\mathcal{T}$ if $\mathcal{M} \equiv_\Sigma \mathcal{T}$;
- a *self-contained* $\equiv_\Sigma$-*module of* $\mathcal{T}$ if $\mathcal{M} \equiv_{\Sigma \cup sig(\mathcal{M})} \mathcal{T}$;
- a *depleting* $\equiv_\Sigma$-*module of* $\mathcal{T}$ if $\emptyset \equiv_{\Sigma \cup sig(\mathcal{M})} \mathcal{T} \setminus \mathcal{M}$.

$\mathcal{M}$ is a *minimal* (*self-contained*, *depleting*) $\equiv_\Sigma$-*module of* $\mathcal{T}$ if $\mathcal{M}$ is a (self-contained, depleting) $\equiv_\Sigma$-module of $\mathcal{T}$, but no proper subset of $\mathcal{M}$ is such a (self-contained, depleting) $\equiv_\Sigma$-module of $\mathcal{T}$.

The main feature of self-contained $\equiv_\Sigma$-modules is that they are indistinguishable from the original TBox not only with respect to Σ but also with respect to their own signature. Such a module is self-contained in the sense that the original TBox does not imply any extra consequences for the module's signature. It follows from the definition that if ≡ satisfies (M_{sig}) then every self-contained $\equiv_\Sigma$-module is also a $\equiv_\Sigma$-module. Depleting modules emphasise a different aspect of modularity: to be a depleting module, it is required that the TBox without the module does not imply any non-tautological consequences for Σ and the module's signature. We will see below that under certain conditions for the inseparability relation this implies being a self-contained module.

However, in general, no non-trivial inclusions between these types of modules exist.

Example 62. (i) If $\mathcal{T}$ is consistent and does not contain symbols from Σ then any $\mathcal{M} \subseteq \mathcal{T}$ is clearly a $\equiv_\Sigma$-module of $\mathcal{T}$, for any inseparability relation ≡ introduced in Section 3, except model inseparability. On the other hand, such $\mathcal{M}$ are not necessarily self-contained (i.e., not necessarily $\equiv_{sig(\mathcal{M})}$-modules of $\mathcal{T}$). Thus, not all $\equiv_\Sigma$-modules are self-contained $\equiv_\Sigma$-modules.

(ii) To show that not all self-contained $\equiv_\Sigma$-modules are depleting $\equiv_\Sigma$-modules, consider $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq B \sqcap B\}$ and $\Sigma = \{A, B\}$. Then $\mathcal{M}_1 = \{A \sqsubseteq B\}$ and $\mathcal{M}_2 = \{A \sqsubseteq B \sqcap B\}$ are self-contained $\equiv_\Sigma$-modules, but $\mathcal{T}$ itself is the only depleting $\equiv_\Sigma$-module of $\mathcal{T}$, for any inseparability relation ≡ introduced in Section 3.

A more interesting example is needed to show that not all depleting $\equiv^c_\Sigma$-modules are (self-contained) $\equiv^c_\Sigma$-modules, where $\equiv^c_\Sigma$ is the Σ-concept inseparability relation in *DL-Lite*$^{\mathcal{N}}_{bool}$.

Example 63. Consider the following modification of Example 6. Let $\Sigma = \{\mathsf{Lecturer}, \mathsf{Course}\}$ and

$$\mathcal{T} = \{\mathsf{Lecturer} \sqsubseteq \exists \mathsf{teaches},\ \exists \mathsf{teaches}^- \sqsubseteq \mathsf{Course},\ \ \mathsf{Course} \sqsubseteq \bot\}.$$

Then $\mathcal{M} = \{\mathsf{Course} \sqsubseteq \bot\}$ is a depleting $\equiv^c_\Sigma$-module of $\mathcal{T}$. However, $\mathcal{M}$ is not a $\equiv^c_\Sigma$-module (and so not a self-contained $\equiv^c_\Sigma$-module) of $\mathcal{T}$ because $\mathcal{T} \models \mathsf{Lecturer} \sqsubseteq \bot$.

Before investigating the relationship between self-contained and depleting modules further, we present a straightforward algorithm extracting *one* minimal $\equiv_\Sigma$-module from a given TBox, using an oracle deciding the inseparability relation $\equiv$.

Theorem 64. *Let $\equiv$ be an inseparability relation in $\mathcal{L}$ satisfying (M_T), $\mathcal{T}$ a TBox in $\mathcal{L}$ and Σ a signature. Then the following algorithm computes a minimal $\equiv_\Sigma$-module of $\mathcal{T}$:*

```
input 𝒯, Σ
ℳ := 𝒯
for each α ∈ ℳ do
  if ℳ \ {α} ≡_Σ ℳ then ℳ := ℳ \ {α}
end for
output ℳ
```

Proof. The algorithm computes a $\equiv_\Sigma$-module $\mathcal{M}$ of $\mathcal{T}$ such that $\mathcal{M} \setminus \{\alpha\}$ is not an $\equiv_\Sigma$-module of $\mathcal{T}$, for any $\alpha \in \mathcal{M}$. By (M_T), no proper subset of such an $\mathcal{M}$ is a $\equiv_\Sigma$-module. ❑

Note that the minimal $\equiv_\Sigma$-module extracted by this algorithm depends on the order of picking the axioms α and that in principle there may be exponentially many distinct minimal $\equiv_\Sigma$-modules of the same TBox.

Example 65. Consider the following generalisation of the TBox from Example 62: for $n < \omega$, let

$$\mathcal{T}_n = \{A_i \sqsubseteq B_i,\ A_i \sqsubseteq B_i \sqcap B_i \mid i \leq n\},$$

and let $\Sigma_n = \{A_i, B_i \mid i \leq n\}$. Then any $\mathcal{M} \subseteq \mathcal{T}$ containing either $A_i \sqsubseteq B_i$ or $A_i \sqsubseteq B_i \sqcap B_i$, for each $i \leq n$, is clearly a minimal $\equiv^c_\Sigma$-module of $\mathcal{T}$, and the number of such modules is 2^n.

We now investigate modules which are induced by inseparability relations that satisfy the replacement property **(replace)** considered in Section 3. For the convenience of the reader, we give the definition again.

Definition 66. An inseparability relation $\equiv$ in $\mathcal{L}$ is *robust under replacement* if, for all TBoxes $\mathcal{T}$, $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\mathcal{L}$ and all signatures Σ, we have $\mathcal{T}_1 \cup \mathcal{T} \equiv_\Sigma \mathcal{T}_2 \cup \mathcal{T}$ whenever $\mathcal{T}_1 \equiv_\Sigma \mathcal{T}_2$ and $sig(\mathcal{T}) \subseteq \Sigma$.

As explained in Section 3, robustness under replacement is fundamental for ontology re-use. Taken together with robustness under vocabulary extensions and having defined the notion of a module, its importance can now be justified in a succinct and precise way. Suppose that an ontology developer imports a $\equiv_\Sigma$-module $\mathcal{M}$ of a TBox $\mathcal{T}$ into her own TBox $\mathcal{O}$. If $\equiv$ is robust under replacement and vocabulary extensions, then $\mathcal{O} \cup \mathcal{T} \equiv_{\Sigma'} \mathcal{O} \cup \mathcal{M}$, for every signature Σ' such that $\Sigma' \cap sig(\mathcal{T}) \subseteq \Sigma$ and $sig(\mathcal{T}) \cap sig(\mathcal{O}) \subseteq \Sigma'$. Thus, these robustness properties ensure that it does not make any difference, as far a such a signature Σ' is concerned, whether she imports the whole $\mathcal{T}$ or some $\equiv_\Sigma$-module $\mathcal{M}$ of $\mathcal{T}$ into $\mathcal{O}$. Moreover, these properties do not depend on $\mathcal{O}$ and can be checked by considering only $\mathcal{T}$ and $\mathcal{M}$.

The following result summarises what has been shown already in Examples 8 and 9 or follows from the corresponding theorems on the equivalence of inseparability notions.

Theorem 67. (i) *The following inseparability relations are <u>not</u> robust under replacement*: *Σ-concept inseparability in DL-Lite*$^{\mathcal{N}}_{bool}$*, and Σ-concept and Σ-query inseparability in DL-Lite*$^{\mathcal{N}}_{horn}$.

(ii) *The following inseparability relations are robust under replacement*: *Σ-query inseparability in DL-Lite*$^{\mathcal{N}}_{bool}$*, strong Σ-concept and strong Σ-query inseparability in DL-Lite*$^{\mathcal{N}}_{bool}$ *and DL-Lite*$^{\mathcal{N}}_{horn}$ *as well as Σ-model inseparability.*

We now give two more reasons explaining the importance of robustness under replacement.

Theorem 68. *If an inseparability relation $\equiv$ in $\mathcal{L}$ is robust under replacement, then every depleting $\equiv_\Sigma$-module is a self-contained $\equiv_\Sigma$-module.*

Proof. If $\mathcal{T} \setminus \mathcal{M} \equiv_{\Sigma \cup sig(\mathcal{M})} \emptyset$, robustness under replacement implies $\mathcal{T} = (\mathcal{T} \setminus \mathcal{M}) \cup \mathcal{M} \equiv_{\Sigma \cup sig(\mathcal{M})} \emptyset \cup \mathcal{M} = \mathcal{M}$. ❑

Thus, the reason why depleting $\equiv^c_\Sigma$-modules are not always self-contained $\equiv^c_\Sigma$-modules (cf. Example 63) is that the inseparability relation $\equiv^c$ is not robust under replacement.

Theorem 69. *Let $\equiv$ be a monotone inseparability relation in $\mathcal{L}$ that is robust under replacement, $\mathcal{T}$ a TBox in $\mathcal{L}$ and Σ a signature. Then there is a unique minimal depleting $\equiv_\Sigma$-module of $\mathcal{T}$, which can be computed by the following algorithm*:

```
input 𝒯, Σ
ℳ := ∅; 𝒲 := ∅
while (𝒯 \ ℳ) ≠ 𝒲 do
  choose α ∈ (𝒯 \ ℳ) \ 𝒲
  𝒲 := 𝒲 ∪ {α}
  if 𝒲 ≢_{Σ∪sig(ℳ)} ∅ then
    ℳ := ℳ ∪ {α}; 𝒲 := ∅
  endif
end while
output ℳ
```

Proof. Let $\mathcal{M}$ be a depleting $\equiv_\Sigma$-module of $\mathcal{T}$, i.e., $\mathcal{T} \setminus \mathcal{M} \equiv_{\Sigma \cup sig(\mathcal{M})} \emptyset$. We first prove the following:

Claim. *For all signatures Σ' with $\Sigma \subseteq \Sigma' \subseteq \Sigma \cup sig(\mathcal{M})$, if $\mathcal{M}_0 \subseteq \mathcal{T}$ is a minimal set with $\mathcal{M}_0 \not\equiv_{\Sigma'} \emptyset$, then $\mathcal{M}_0 \subseteq \mathcal{M}$.*

Proof of claim. Suppose the claim does not hold, i.e., $\mathcal{M}_0 \not\subseteq \mathcal{M}$. Then we must have $\mathcal{X} \equiv_{\Sigma'} \emptyset$, where $\mathcal{X} = \mathcal{M} \cap \mathcal{M}_0$ (for otherwise $\mathcal{X}$ is a proper subset of $\mathcal{M}_0$ with $\mathcal{X} \not\equiv_{\Sigma'} \emptyset$, contrary to the minimality of $\mathcal{M}_0$). As $\mathcal{M}$ is a depleting Σ-module and $sig(\mathcal{X}) \subseteq sig(\mathcal{M})$, by robustness under replacement, $(\mathcal{T} \setminus \mathcal{M}) \cup \mathcal{X} \equiv_{\Sigma \cup sig(\mathcal{M})} \mathcal{X}$. Using $\Sigma' \subseteq \Sigma \cup sig(\mathcal{M})$, (M_{sig}), and transitivity of $\equiv_{\Sigma'}$, we obtain from $\mathcal{X} \equiv_{\Sigma'} \emptyset$ that $(\mathcal{T} \setminus \mathcal{M}) \cup \mathcal{X} \equiv_{\Sigma'} \emptyset$. By $(M_\top)$, we obtain from $\emptyset \subseteq \mathcal{M}_0 \subseteq (\mathcal{T} \setminus \mathcal{M}) \cup \mathcal{X}$ that $\mathcal{M}_0 \equiv_{\Sigma'} \emptyset$, which is a contradiction.

Using this claim, one can easily prove by induction that each $\mathcal{M}$ computed during a run of the algorithm of Theorem 69 on input $\mathcal{T}$ and Σ is contained in every depleting $\equiv_\Sigma$-module of $\mathcal{T}$. Hence, its output $\mathcal{M}$ is contained in every depleting $\equiv_\Sigma$-module of $\mathcal{T}$. On the other hand, by the termination condition of the algorithm, this $\mathcal{M}$ is a depleting $\equiv_\Sigma$-module of $\mathcal{T}$. Consequently, $\mathcal{M}$ is the uniquely determinable minimal depleting $\equiv_\Sigma$-module of $\mathcal{T}$. ❑

The algorithm above computes *the* minimal depleting $\equiv_\Sigma$-module in quadratic time by calling the oracle deciding the inseparability relation $\equiv$ at most $|\mathcal{T}|^2$ times.

It follows that minimal depleting modules have the advantage of being uniquely determined (under mild conditions), which sharply contrasts with the behaviour of the other types of modules. Another advantage is that depleting modules support modular ontology development in the following sense. Suppose $\mathcal{M}$ is a depleting $\equiv_\Sigma$-module of $\mathcal{T}$ and $\equiv$ is robust under replacement and vocabulary extensions. Then one can import into the ontology $\mathcal{T} \setminus \mathcal{M}$ any module $\mathcal{M}'$ such that $sig(\mathcal{M}') \cap sig(\mathcal{T}) \subseteq \Sigma \cup sig(\mathcal{M})$ and be sure that $\mathcal{T} \setminus \mathcal{M}$ does not interfere with $\mathcal{M}'$—i.e., $(\mathcal{T} \setminus \mathcal{M}) \cup \mathcal{M}' \equiv_{\Sigma'} \mathcal{M}'$ whenever $\Sigma' \cap sig(\mathcal{T} \setminus \mathcal{M}) \subseteq \Sigma \cup sig(\mathcal{M})$. The importance of this property was first pointed out in [17].

In the following illustrative example we compute three kinds of modules in *DL-Lite*$^{\mathcal{N}}_{bool}$:

– minimal Σ-concept inseparability modules (MCM),

– minimal Σ-query inseparability modules (MQM), and

– minimal depleting Σ-query inseparability modules (MDQM).

These abbreviations will be also used in Section 9.

Example 70. Consider the following *DL-Lite*$^{\mathcal{N}}_{bool}$ TBox $\mathcal{T}$:

(1) $\mathsf{Publisher} \sqsubseteq \exists \mathsf{pubHasDistrib}$
(2) $\exists \mathsf{pubHasDistrib}^- \sqsubseteq \mathsf{Distributor}$
(3) $\mathsf{Publisher} \sqsubseteq \neg \mathsf{Distributor}$
(4) $\exists \mathsf{pubHasDistrib} \sqsubseteq \mathsf{Publisher}$
(5) $\mathsf{Publisher} \sqsubseteq \leq 1\, \mathsf{pubHasDistrib}$
(6) $\mathsf{Role} \sqsubseteq \neg \mathsf{Distributor}$
(7) $\mathsf{User} \sqsubseteq \neg \mathsf{Distributor}$
(8) $\mathsf{Publisher} \sqsubseteq \exists \mathsf{pubAdmedBy}$
(9) $\exists \mathsf{pubAdmedBy}^- \sqsubseteq \mathsf{AdmUser} \sqcup \mathsf{BookUser}$
(10) $\mathsf{AdmUser} \sqsubseteq \mathsf{User}$
(11) $\mathsf{BookUser} \sqsubseteq \mathsf{User}$
(12) $\mathsf{User} \sqsubseteq \exists \mathsf{hasRole}$
(13) $\exists \mathsf{hasRole}^- \sqsubseteq \mathsf{Role}$
(14) $\mathsf{Role} \sqsubseteq \neg \mathsf{Publisher}$
(15) $\mathsf{User} \sqsubseteq \neg \mathsf{Publisher}$
(16) $\mathsf{Role} \sqsubseteq \neg \mathsf{User}$
(17) $\mathsf{User} \sqsubseteq \exists \mathsf{userAdmedBy}$
(18) $\exists \mathsf{userAdmedBy}^- \sqsubseteq \mathsf{AdmUser}$
(19) $\exists \mathsf{userAdmedBy} \sqsubseteq \mathsf{User}$
(20) $\exists \mathsf{pubAdmedBy} \sqsubseteq \mathsf{Publisher}$

(which is part of the larger Core ontology to be discussed in Section 9), and let $\Sigma = \{\mathsf{Publisher}\}$. Observe that the MCM of $\mathcal{T}$ is empty, which is typical of singleton signatures and Σ-concept inseparability, as no interesting concept inclusions over a singleton signature exist. In contrast, there are three different MQMs of $\mathcal{T}$:

$$\mathcal{M}_D = \{(1), (2), (3)\}, \qquad \mathcal{M}_R = \{(8), (9), (10), (11), (12), (13), (14)\}, \qquad \mathcal{M}_U = \{(8), (9), (10), (11), (15)\}.$$

First, they are indeed Σ-query inseparable from $\mathcal{T}$, which can be verified via the semantic criterion of Theorem 21. Second, they are minimal. For consider the ABox $\mathcal{A} = \{\mathsf{Publisher}(a)\}$ and the query $\mathsf{q} = \exists x\, \neg \mathsf{Publisher}(x)$. Clearly, we have $(\mathcal{T}, \mathcal{A}) \models \mathsf{q}$, while $(\mathcal{T}', \mathcal{A}) \not\models \mathsf{q}$, for any proper subset $\mathcal{T}'$ of $\mathcal{M}_D$, $\mathcal{M}_R$ or $\mathcal{M}_U$. In contrast to this finding, *the* MDQM of $\mathcal{T}$ is $\mathcal{T}$ itself. To illustrate that this is indeed the case, consider the TBox $\mathcal{T}'$ with axioms (17)–(19) and show that $\mathcal{M} = \mathcal{T} \setminus \mathcal{T}'$ is not a MDQM of $\mathcal{T}$. In other words, we show that $\mathcal{T}'$ is $sig(\mathcal{M})$-query separable from $\emptyset$. (Note that $sig(\mathcal{M}) = sig(\mathcal{T}) \setminus \{\mathsf{userAdmedBy}\}$.) Let $\mathcal{A} = \{\mathsf{User}(a)\}$ and $\mathsf{q} = \exists x\, \mathsf{AdmUser}(x)$. Then clearly $(\mathcal{T}', \mathcal{A}) \models \mathsf{q}$.

Consider now $\Sigma' = \{\mathsf{Publisher}, \mathsf{pubHasDistrib}\}$. Then the only MCM-module of $\mathcal{T}$ with respect to Σ' consists of axioms (1)–(5), and there are two MQMs with respect to Σ':

$$\mathcal{M}'_R = \mathcal{M}_D \cup \mathcal{M}_R \cup \{(4), (5), (6)\} \qquad \text{and} \qquad \mathcal{M}'_U = \mathcal{M}_D \cup \mathcal{M}_U \cup \{(4), (5), (7)\}.$$

8. Forgetting and uniform interpolation

When extracting a subset $\mathcal{M}$ from an ontology $\mathcal{T}$ that 'says the same about a signature Σ' as $\mathcal{T}$, one typically has to include into $\mathcal{M}$ a large number of axioms from $\mathcal{T}$ that contain non-Σ-symbols. Example 70 above shows that even for Σ of size one or two, many additional symbols occur in the module. In this section, we aim at 'extracting' new ontologies from a given ontology that 'say the same about Σ' as the original ontology and, in addition, do not use non-Σ-symbols. Often, this can only be achieved by introducing new axioms that do not occur in the original ontology. In mathematical logic parlance such an ontology would be called a uniform interpolant of the original ontology [22, 23], whereas in artificial intelligence, computing the new ontology is known as *forgetting* (the non-Σ-symbols) [19, 20, 27]. The advantage of forgetting over module extraction is that it does not depend on the way the original ontology is formulated: whereas modules are subsets of the original ontology, forgetting can (and will) be defined independently from the axiomatisation of the ontology. Of course, this can also be regarded as a disadvantage because the ontology engineer is not familiar with the new axioms, which can be hard to understand and process.

To formalise forgetting/uniform interpolation, we employ again our notions of Σ-inseparability and say that a TBox $\mathcal{T}_\Sigma$ is a uniform interpolant of a TBox $\mathcal{T}$ with respect to Σ if the signature of $\mathcal{T}_\Sigma$ is included in Σ and $\mathcal{T}$ and $\mathcal{T}_\Sigma$ are Σ-inseparable. Of course, the problems whether such a TBox $\mathcal{T}_\Sigma$ exists, its size, and whether it can be constructed effectively, depend on the available language constructs, the signature Σ, and the type of Σ-inseparability one is interested in. In this section, we consider the notions of uniform interpolation corresponding to Σ-concept inseparability in *DL-Lite*$^{\mathcal{N}}_{horn}$ and *DL-Lite*$^{\mathcal{N}}_{bool}$ and Σ-query inseparability in *DL-Lite*$^{\mathcal{N}}_{bool}$. A more systematic study of how inseparability relations can be used to define forgetting is beyond the scope of this paper (see [26] for such a study for extensions of the description logic $\mathcal{EL}$).

We start by defining forgetting and uniform interpolation based on Σ-concept inseparability in *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$.

Definition 71. Let $\mathcal{L} \in \{\textit{DL-Lite}^{\mathcal{N}}_{horn}, \textit{DL-Lite}^{\mathcal{N}}_{bool}\}$. We say that $\mathcal{L}$ *admits forgetting* (or *has uniform interpolation*) if, for every TBox $\mathcal{T}$ in $\mathcal{L}$ and every signature Σ, there exists a TBox $\mathcal{T}_\Sigma$ in $\mathcal{L}$ with $sig(\mathcal{T}_\Sigma) \subseteq \Sigma$ such that $\mathcal{T}$ and $\mathcal{T}_\Sigma$ are Σ'-concept inseparable in $\mathcal{L}$ for all Σ' with $sig(\mathcal{T}) \cap \Sigma' \subseteq \Sigma$. In this case, $\mathcal{T}_\Sigma$ is called a *uniform interpolant* of $\mathcal{T}$ with respect to Σ in $\mathcal{L}$.

Note that this definition appears to be more restrictive than what was indicated in the informal discussion above: instead of demanding that $\mathcal{T}_\Sigma$ and $\mathcal{T}$ are Σ-concept inseparable, we require that $\mathcal{T}_\Sigma$ and $\mathcal{T}$ are Σ'-concept inseparable *for every* Σ' *with* $sig(\mathcal{T}) \cap \Sigma' \subseteq \Sigma$. It is readily seen, however, that the two definitions are actually equivalent because Σ-concept entailment is robust under vocabulary extensions.

Example 72. Let $\mathcal{T} = \{\mathsf{Hand} \sqsubseteq \mathsf{BodyPart},\ \mathsf{BodyPart} \sqsubseteq \mathsf{PhysicalObject}\}$ and $\Sigma = \{\mathsf{Hand}, \mathsf{PhysicalObject}\}$. Then the TBox $\mathcal{T}_\Sigma = \{\mathsf{Hand} \sqsubseteq \mathsf{PhysicalObject}\}$ is a uniform interpolant of $\mathcal{T}$ with respect to Σ in both $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and $\textit{DL-Lite}^{\mathcal{N}}_{horn}$.

Note that if $\mathcal{L}$ has uniform interpolation, then in principle we can use uniform interpolants to check Σ-concept entailment in $\mathcal{L}$. Indeed, suppose that we are given TBoxes $\mathcal{T}$ and $\mathcal{T}'$ in $\mathcal{L}$ and that we want to see whether $\mathcal{T}$ Σ-concept entails $\mathcal{T}'$ in $\mathcal{L}$. To this end, we compute a uniform interpolant $\mathcal{T}'_\Sigma$ of $\mathcal{T}'$ with respect to Σ in $\mathcal{L}$. And then we have the following:

– $\mathcal{T}$ Σ-concept entails $\mathcal{T}'$ if, and only if, $\mathcal{T} \models C_1 \sqsubseteq C_2$, for all $(C_1 \sqsubseteq C_2) \in \mathcal{T}'_\Sigma$.

Thus, checking Σ-concept entailment can be reduced to computing uniform interpolants and checking subsumption in $\mathcal{L}$. The following theorem states that $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and $\textit{DL-Lite}^{\mathcal{N}}_{horn}$ do enjoy uniform interpolation. (It**!** will be proved in Section A.1 of the appendix and subsequently used to establish some results stated earlier in this paper.) moved

Theorem 73. *Let* $\mathcal{L} \in \{\textit{DL-Lite}^{\mathcal{N}}_{bool}, \textit{DL-Lite}^{\mathcal{N}}_{horn}\}$. *Then* $\mathcal{L}$ *has uniform interpolation, and a uniform interpolant of a TBox* $\mathcal{T}$ *with respect to* Σ *in* $\mathcal{L}$ *can be constructed effectively.*

In the worst case, the uniform interpolants given in the proof of this theorem are exponential**!** in the size of $\mathcal{T}$. changed
Note that even in propositional logic all known algorithms for computing uniform interpolants return, in the worst case, interpolants of exponential size. In fact, it is known that, unless P = NC (which is regarded as rather unlikely), there do not always exist uniform interpolants of polynomial size in propositional logic [54].

Example 74. For $\textit{DL-Lite}^{\mathcal{N}}_{horn}$, one can give a simple example showing that minimal uniform interpolants are, in the worst case, of exponential size. Indeed, let

$$\mathcal{T}_n = \{A \equiv B_1 \sqcap \cdots \sqcap B_n\} \cup \{A^i_j \sqsubseteq B_i \mid 1 \le i \le n, j = 1, 2\} \quad \text{and} \quad \Sigma_n = \{A\} \cup \{A^i_j \mid 1 \le i \le n, j = 1, 2\}.$$

Then

$$\mathcal{T}_{\Sigma_n} = \{A^1_{j_1} \sqcap \cdots \sqcap A^n_{j_n} \sqsubseteq A \mid 1 \le j_1, \ldots, j_n \le 2\}$$

is a uniform interpolant of $\mathcal{T}_n$ with respect to Σ_n in $\textit{DL-Lite}^{\mathcal{N}}_{horn}$. It is of size 2^n, and there is no smaller uniform interpolant in $\textit{DL-Lite}^{\mathcal{N}}_{horn}$. It is worth mentioning, however, that there exists a uniform interpolant of $\mathcal{T}_n$ with respect to Σ_n in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ which is of *polynomial* size:

$$\mathcal{T}'_{\Sigma_n} = \{ \prod_{1 \le i \le n} (A^i_1 \sqcup A^i_2) \sqsubseteq A\}.$$

Of course, these worst case lower bounds do not imply that in practice it is unfeasible to compute uniform interpolants; for example, it would be interesting to conduct experiments on deciding Σ-concept entailment using Theorem 73 and compare the performance of this approach with the one based on the QBF encoding to be discussed below. For experimental results on computing uniform interpolants for TBoxes in the description logic $\mathcal{EL}$ we refer the reader to [26].

The notion of uniform interpolation considered above reflects the interpretation of 'saying the same about a vocabulary Σ' as being Σ-concept inseparable. How can this notion of uniform interpolation be modified when we are interested not in Σ-concept inseparability but, say, in Σ-query inseparability? The straightforward modification of Definition 71 by replacing concept inseparability with query inseparability (or any other notion of inseparability introduced above) is unsatisfactory, as shown by the following example.

Example 75. Let $\mathcal{L} \in \{\textit{DL-Lite}^{\mathcal{N}}_{bool}, \textit{DL-Lite}^{\mathcal{N}}_{horn}\}$. Consider the TBox

$$\mathcal{T} = \{\, \mathsf{Lecturer} \sqsubseteq \exists \mathsf{teaches},\ \exists \mathsf{teaches}^{-} \sqsubseteq \mathsf{Course} \,\}$$

from Example 6, and let $\Sigma = \{\mathsf{Lecturer}, \mathsf{Course}\}$. Then $\mathcal{T}_\Sigma = \emptyset$ is a uniform interpolant of $\mathcal{T}$ with respect to Σ in $\mathcal{L}$ because, as we have already seen, the empty TBox and $\mathcal{T}$ are Σ-concept inseparable in $\mathcal{L}$. Equivalently, we know that

$$\{\, C \sqsubseteq D \in \textit{DL-Lite}^{\mathcal{N}}_{bool} \mid \mathcal{T} \models C \sqsubseteq D,\ \textit{sig}(C \sqsubseteq D) \in \Sigma \,\}$$

consists of tautologies only. We also know that no Σ-TBox consisting of tautologies is Σ-query inseparable in $\mathcal{L}$ from $\mathcal{T}$. Thus, there does not exist a Σ-TBox $\mathcal{T}'_\Sigma$ in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ such that $\mathcal{T}'_\Sigma$ and $\mathcal{T}$ are Σ-query inseparable. Hence, the straightforward modification of Definition 71 by replacing concept inseparability with query inseparability leads to a definition which allows very simple TBoxes in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and signatures Σ without uniform interpolants.

The example above shows that to obtain a satisfactory notion of forgetting and uniform interpolation that reflects Σ-query inseparability, apart from replacing Σ-concept inseparability with Σ-query inseparability in Definition 71, we also have to increase the expressive power of the description logic in which uniform interpolants are axiomatised.

Denote by $\textit{DL-Lite}^{u}_{bool}$ the extension of $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ with the *universal role* $\mathbf{U}$, where the $\textit{DL-Lite}^{u}_{bool}$ *concepts* are defined as follows

$$D \quad ::= \quad C \quad | \quad \exists \mathbf{U}.C \quad | \quad D_1 \sqcap D_2 \quad | \quad \neg D$$

where C are $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ concepts. Given an interpretation $\mathcal{I}$, we set $(\exists \mathbf{U}.C)^{\mathcal{I}} = \Delta^{\mathcal{I}}$ if $C^{\mathcal{I}} \neq \emptyset$, and $(\exists \mathbf{U}.C)^{\mathcal{I}} = \emptyset$ otherwise. The remaining model-theoretic notions are defined exactly as for $\textit{DL-Lite}^{\mathcal{N}}_{bool}$. Using the construction from [9] one can show that the subsumption problem '$\mathcal{T} \models C_1 \sqsubseteq C_2$?' is still coNP-complete for TBoxes $\mathcal{T}$ in $\textit{DL-Lite}^{u}_{bool}$ and concept inclusions $C_1 \sqsubseteq C_2$ in $\textit{DL-Lite}^{u}_{bool}$. It is important that we regard $\mathbf{U}$ as a *logical symbol*, so that $\textit{sig}(\exists \mathbf{U}.C) = \textit{sig}(C)$.

Definition 76. Let $\mathcal{T}$ be a TBox in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and Σ a signature. A TBox $\mathcal{T}_\Sigma$ in $\textit{DL-Lite}^{u}_{bool}$ is called a *uniform query interpolant* of $\mathcal{T}$ with respect to Σ in $\textit{DL-Lite}^{u}_{bool}$ if $\mathcal{T} \models \alpha$ for all $\alpha \in \mathcal{T}_\Sigma$, $\textit{sig}(\mathcal{T}_\Sigma) \subseteq \Sigma$, and $\mathcal{T}$ and $\mathcal{T}_\Sigma$ are Σ-query inseparable.

Example 77. Consider again the TBox

$$\mathcal{T} = \{\, \mathsf{Lecturer} \sqsubseteq \exists \mathsf{teaches},\ \exists \mathsf{teaches}^{-} \sqsubseteq \mathsf{Course} \,\} \quad \text{and} \quad \Sigma = \{\mathsf{Lecturer}, \mathsf{Course}\}.$$

Then $\mathcal{T}_\Sigma = \{\mathsf{Lecturer} \sqsubseteq \exists \mathbf{U}.\mathsf{Course}\}$ is a uniform query interpolant of $\mathcal{T}$ with respect to Σ in $\textit{DL-Lite}^{u}_{bool}$.

To analyse and justify this definition of uniform query interpolants, we first show that one can use uniform query interpolants to understand Σ-query entailment in the same way as uniform interpolants can be used to understand Σ-concept entailment.

Theorem 78. *Let $\mathcal{T}$ and $\mathcal{T}'$ be TBoxes in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and Σ a signature. And let $\mathcal{T}'_\Sigma$ be a uniform query interpolant of $\mathcal{T}'$ with respect to Σ in $\textit{DL-Lite}^{u}_{bool}$. Then $\mathcal{T}$ Σ-query entails $\mathcal{T}'$ if, and only if, $\mathcal{T} \models C_1 \sqsubseteq C_2$, for every $(C_1 \sqsubseteq C_2) \in \mathcal{T}'_\Sigma$.*

Finally, one can show that uniform query interpolants always exist.

Theorem 79. *For every TBox $\mathcal{T}$ in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and every signature Σ, one can construct a uniform query interpolant $\mathcal{T}_\Sigma$ of $\mathcal{T}$ with respect to Σ in $\textit{DL-Lite}^{u}_{bool}$.*

We close this section with a brief discussion of open problems and related work on forgetting and uniform interpolation. We have proposed notions of forgetting and uniform interpolation induced by concept inseparability in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ and $\textit{DL-Lite}^{\mathcal{N}}_{horn}$ as well as query inseparability in $\textit{DL-Lite}^{\mathcal{N}}_{bool}$. We have seen that the latter case is not straightforward, as we had to enrich the underlying description logic $\textit{DL-Lite}^{\mathcal{N}}_{bool}$ by the universal role so as to obtain a notion for which query uniform interpolants always exist. Developing uniform interpolants based on the remaining Σ-inseparability relations is an interesting problem, but it goes beyond the scope of this paper.

Forgetting concepts (but not roles) in *DL-Lite* was studied in [27] using a resolution-based technique. It is also worth mentioning that, for many standard DLs such as $\mathcal{ALC}$ and even $\mathcal{EL}$, uniform interpolants do not always exist [24, 25, 26].

9. Experimental results

In order to see whether the logic-based approach to checking inseparability relations between and extracting minimal modules from *DL-Lite* ontologies is feasible in practice, we conducted a series of experiments with a number of 'typical' medium-size *DL-Lite*$^{\mathcal{N}}_{bool}$ ontologies. Instead of developing and implementing algorithms for checking the Σ-concept and Σ-query entailment criteria of Theorems 20, 21 and 47, we encoded these criteria by means of quantified Boolean formulas (QBFs, for short) and then employed standard *off-the-shelf general purpose* QBF solvers. In this section, we discuss some details of the encodings and the results of the experiments.

9.1. QBF encodings

We begin by showing how the conditions of Proposition 46 for (precise) realisability of sets of types can be encoded by means of QBFs. To represent ΣQ-types, we will use *ΣQ-type variables* $\boldsymbol{b}$, which are lists of propositional variables of the form $B_{\boldsymbol{b}}$, where B is a basic ΣQ-concept different from $\bot$ and $\top$. This notation is extended then inductively to arbitrary concepts built from basic ΣQ-concepts: given a *ΣQ-type variable* $\boldsymbol{b}$, we set

$$\bot_{\boldsymbol{b}} = \bot, \qquad \top_{\boldsymbol{b}} = \top, \qquad (\neg C)_{\boldsymbol{b}} = \neg C_{\boldsymbol{b}}, \qquad (C_1 \sqcap C_2)_{\boldsymbol{b}} = (C_1)_{\boldsymbol{b}} \wedge (C_2)_{\boldsymbol{b}}.$$

Let $\mathcal{T}$ be a TBox in *DL-Lite*$^{\mathcal{N}}_{bool}$, m the number of role names in $\mathcal{T}$, and $Q \supseteq Q_{\mathcal{T}}$. To encode the notion of a $\mathcal{T}$-witness for a set of $(k+1)$-many $sig(\mathcal{T})Q$-types, which is used in Proposition 46, we take $sig(\mathcal{T})Q$-type variables $\boldsymbol{b}_0, \dots, \boldsymbol{b}_{k+2m}$ and consider the propositional formula

$$\Phi^k_{\mathcal{T}}(\boldsymbol{b}_0, \dots, \boldsymbol{b}_{k+2m}) \;=\; \bigwedge_{j=0}^{k+2m} \bigwedge_{C_1 \sqsubseteq C_2 \in \mathcal{T}} ((C_1)_{\boldsymbol{b}_j} \to (C_2)_{\boldsymbol{b}_j}) \quad \wedge \quad \bigwedge_{i=1}^{m} \Big[\Big(\bigvee_{j=0}^{k+2m} (\exists P_i^-)_{\boldsymbol{b}_j} \to (\exists P_i)_{\boldsymbol{b}_{k+2i-1}} \Big) \wedge \Big(\bigvee_{j=0}^{k+2m} (\exists P_i)_{\boldsymbol{b}_j} \to (\exists P_i^-)_{\boldsymbol{b}_{k+2i}} \Big) \Big].$$

The meaning of this formula, encoding conditions $(\mathbf{w}_2)$ and $(\mathbf{w}_3)$ of Definition 45, is explained by the following proposition:

Proposition 80. *A pair $\Xi^{\mathcal{T}} = ((\boldsymbol{t}_0, \dots, \boldsymbol{t}_k), (\boldsymbol{t}_{k+1}, \dots, \boldsymbol{t}_{k+2m}))$ of sequences of $sig(\mathcal{T})Q$-types is a $\mathcal{T}$-witness for a set $\Xi = \{\boldsymbol{t}_0, \dots, \boldsymbol{t}_k\}$ of $sig(\mathcal{T})Q$-types if, and only if, $\Phi^k_{\mathcal{T}}(\boldsymbol{b}_0, \dots, \boldsymbol{b}_{k+2m})$ is true under the following assignment $\mathfrak{a}$ of the truth-values to the propositional variables $B_{\boldsymbol{b}_j}$, for all basic $sig(\mathcal{T})Q$-concepts B and $0 \le j \le k + 2m$:*

$$\mathfrak{a}(B_{\boldsymbol{b}_j}) = \mathsf{T} \quad \textit{if, and only if,} \quad B \in \boldsymbol{t}_j.$$

Using the formulas $\Phi^k_{\mathcal{T}}(\boldsymbol{b}_0, \dots, \boldsymbol{b}_{k+2m})$ and Propositions 46 and 80, we can now represent the problem of deciding whether $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ as the truth problem for certain QBFs.

Let m_i be the number of role names in $\mathcal{T}_i$, $i = 1, 2$, and $Q = Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$. Take, a ΣQ-type variable $\boldsymbol{b}$, a $(sig(\mathcal{T}_i) \setminus \Sigma)Q$-type variable $\widehat{\boldsymbol{b}^i_0}$ and $sig(\mathcal{T}_i)Q$-type variables $\boldsymbol{b}^i_1, \dots, \boldsymbol{b}^i_{2m_i}$, for $i = 1, 2$, and consider the following QBF:

$$\forall\, \boldsymbol{b} \quad \Big[\exists\, \widehat{\boldsymbol{b}^1_0}, \boldsymbol{b}^1_1, \dots, \boldsymbol{b}^1_{2m_1} \quad \Phi^1_{\mathcal{T}_1}(\boldsymbol{b} \cdot \widehat{\boldsymbol{b}^1_0}, \boldsymbol{b}^1_1, \dots, \boldsymbol{b}^1_{2m_1}) \quad \to \quad \exists\, \widehat{\boldsymbol{b}^2_0}, \boldsymbol{b}^2_1, \dots, \boldsymbol{b}^2_{2m_2} \quad \Phi^1_{\mathcal{T}_2}(\boldsymbol{b} \cdot \widehat{\boldsymbol{b}^2_0}, \boldsymbol{b}^2_1, \dots, \boldsymbol{b}^2_{2m_2}) \Big], \quad (3)$$

where $\boldsymbol{b} \cdot \widehat{\boldsymbol{b}^i_0}$ is the $sig(\mathcal{T}_i)Q$-type variable which is the concatenation of $\boldsymbol{b}$ and $\widehat{\boldsymbol{b}^i_0}$, $i = 1, 2$. Informally, this QBF says the following: for every ΣQ-type $\boldsymbol{t}$ (represented by $\boldsymbol{b}$), if $\boldsymbol{t}$ can be extended to a $sig(\mathcal{T}_1)Q$-type (by means of $\widehat{\boldsymbol{b}^1_0}$) for which there exist $2m_1$-many $sig(\mathcal{T}_1)Q$-types (represented by $\boldsymbol{b}^1_1, \dots, \boldsymbol{b}^1_{2m_1}$) such that the resulting set of $2m_1 + 1$ types is $\mathcal{T}_1$-realisable (as stated by $\Phi^1_{\mathcal{T}_1}(\boldsymbol{b} \cdot \widehat{\boldsymbol{b}^1_0}, \boldsymbol{b}^1_1, \dots, \boldsymbol{b}^1_{2m_1})$), then $\boldsymbol{t}$ can be also extended to a $sig(\mathcal{T}_2)Q$-type for which $2m_2$-many $sig(\mathcal{T}_2)Q$-types can be found such that the resulting set of these $2m_2 + 1$ types is $\mathcal{T}_2$-realisable. In other words, QBF (3) is true if, and only if, $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ (in *DL-Lite*$^{\mathcal{N}}_{bool}$ or *DL-Lite*$^{\mathcal{N}}_{horn}$). Note, by the way, that together with the known results on the complexity of classes of QBFs [49, 55], this encoding provides an alternative proof of the upper complexity bounds deciding Σ-concept entailment for both *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$.

ResearchStaff $\sqcap$ Visiting $\sqsubseteq$ $\bot$	Academic $\sqsubseteq$ $\exists$teaches $\sqcap$ $\neg \geq 2$ teaches
$\exists$teaches $\sqsubseteq$ Academic $\sqcup$ ResearchStaff	$\exists$writes $\sqsubseteq$ Academic $\sqcup$ ResearchStaff
ResearchStaff $\sqsubseteq$ $\exists$worksIn	$\exists$worksIn$^-$ $\sqsubseteq$ Project
Project $\sqsubseteq$ $\exists$manages$^-$	$\exists$manages $\sqsubseteq$ Academic $\sqcup$ Visiting

Figure 3: A fragment $\mathcal{T}_1$ of the 'department ontology'.

In a similar manner we can encode the criterion of Theorem 47 for Σ-query entailment between *DL-Lite*$^{\mathcal{N}}_{bool}$ TBoxes. Take ΣQ-type variables $\boldsymbol{b}_0, \dots, \boldsymbol{b}_{2m_1}$, $(sig(\mathcal{T}_i) \setminus \Sigma)Q$-type variables $\widehat{\boldsymbol{b}^i_0}, \dots, \widehat{\boldsymbol{b}^i_{2m_1}}$, for $i = 1, 2$, and $sig(\mathcal{T}_2)Q$-type variables $\boldsymbol{b}^3_1, \dots, \boldsymbol{b}^3_{2m_2}$, and consider the QBF

$$\forall\, \boldsymbol{b}_0, \dots, \boldsymbol{b}_{2m_1} \Big[\exists\, \widehat{\boldsymbol{b}^1_0}, \dots, \widehat{\boldsymbol{b}^1_{2m_1}} \;\; \Phi^1_{\mathcal{T}_1}(\boldsymbol{b}_0 \cdot \widehat{\boldsymbol{b}^1_0}, \dots, \boldsymbol{b}_{2m_1} \cdot \widehat{\boldsymbol{b}^1_{2m_1}}) \;\; \rightarrow$$
$$\exists\, \widehat{\boldsymbol{b}^2_0}, \dots, \widehat{\boldsymbol{b}^2_{2m_1}} \; \exists\, \boldsymbol{b}^3_1, \dots, \boldsymbol{b}^3_{2m_2} \left(\Phi^{1+2m_1}_{\mathcal{T}_2}(\boldsymbol{b}_0 \cdot \widehat{\boldsymbol{b}^2_0} \dots, \boldsymbol{b}_{2m_1} \cdot \widehat{\boldsymbol{b}^2_{2m_1}}, \boldsymbol{b}^3_1, \dots, \boldsymbol{b}^3_{2m_2}) \wedge \bigwedge_{i=1}^{2m_2} \bigvee_{j=0}^{2m_1} \chi(\boldsymbol{b}^3_i, \boldsymbol{b}_j) \right) \Big], \quad (4)$$

where $\chi(\boldsymbol{b}^3_i, \boldsymbol{b}_j)$ is the conjunction of equivalences $(B_{\boldsymbol{b}^3_i} \leftrightarrow B_{\boldsymbol{b}_j})$, for all basic ΣQ-concepts (which is required to encode condition **(w-pr)** of Definition 45). It can be seen that QBF (4) is true if, and only if, $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in *DL-Lite*$^{\mathcal{N}}_{bool}$.

Although the QBFs (3) and (4) look similar, belong to the same class of $\forall\exists$ QBFs and the same complexity class, in practice they behave quite differently. In (3), we take a ΣQ-type $\boldsymbol{t}$, (i) extend $\boldsymbol{t}$ to a $sig(\mathcal{T}_1)Q$-type, (ii) check whether there are 'witnesses' for all the roles in that type and the types providing those witnesses, and if this is the case, we repeat (i) and (ii) again for $\mathcal{T}_2$ in place of $\mathcal{T}_1$. QBF (4) is much more complex not only because now we have to start with a set of $(1 + 2m_1)$ ΣQ-types rather than a single type. More importantly, the $\mathcal{T}_2$-witnesses we choose for these types are not *arbitrary* but must have Σ-restrictions that coincide with some of the original $(1 + 2m_1)$ ΣQ-types. This last condition, represented by a formula with $2m_2 \times (1 + 2m_1)$-many occurrences of $\chi(\boldsymbol{b}^3_i, \boldsymbol{b}_j)$, makes QBF (4) computationally much more costly in practice.

9.2. Experiments with Σ-entailment

To evaluate the performance of QBF solvers when checking Σ-concept and Σ-query entailment, we used an extension of the *DL-Lite*$^{\mathcal{N}}_{bool}$ approximation of the standard 'department ontology' (cf. `http://swat.cse.lehigh.edu/projects/lubm/`). The reader can appreciate the complexity of the problems the QBF solvers were facing by trying to check whether the ontology $\mathcal{T}_1$ in Fig. 3 (which is a tiny part of our department ontology) Σ-concept and Σ-query entails the ontology $\mathcal{T}_2 = \mathcal{T}_1 \cup \{\mathsf{Visiting} \sqsubseteq \geq 2\,\mathsf{writes}\}$, for $\Sigma = \{\mathsf{teaches}\}$.

As our benchmarks, we considered three series of instances of the form $(\mathcal{T}_1, \mathcal{T}_2, \Sigma)$. In the *NN-series*, $\mathcal{T}_1$ does not Σ-concept entails $\mathcal{T}_2$; in the *YN-series*, $\mathcal{T}_1$ Σ-concept but not Σ-query entails $\mathcal{T}_2$; and in the *YY-series*, $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$. The sizes of the instances are uniformly distributed over the intervals given in the table below:

	no. of	no. of axioms		no. of basic concepts		
series	instances	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_1$	$\mathcal{T}_2$	Σ
NN	840	59–308	74–396	47–250	49–300	5–103
YN	504	56–302	77–382	44–246	58–298	6–89
YY	624	43–178	43–222	40–158	40–188	5–64

The next table illustrates the sizes of the QBF translations of our instances for both Σ-concept and Σ-query entailment:

	Σ-concept entailment QBF		Σ-query entailment QBF	
series	no. of variables	no. of clauses	no. of variables	no. of clauses
NN	1,469–48,631	2,391–74,621	1,715–60,499	5,763–1,217,151
YN	1,460–46,873	2,352–71,177	1,755–59,397	7,006–1,122,361
YY	1,006–16,033	1,420–23,363	1,202–20,513	2,963–204,889

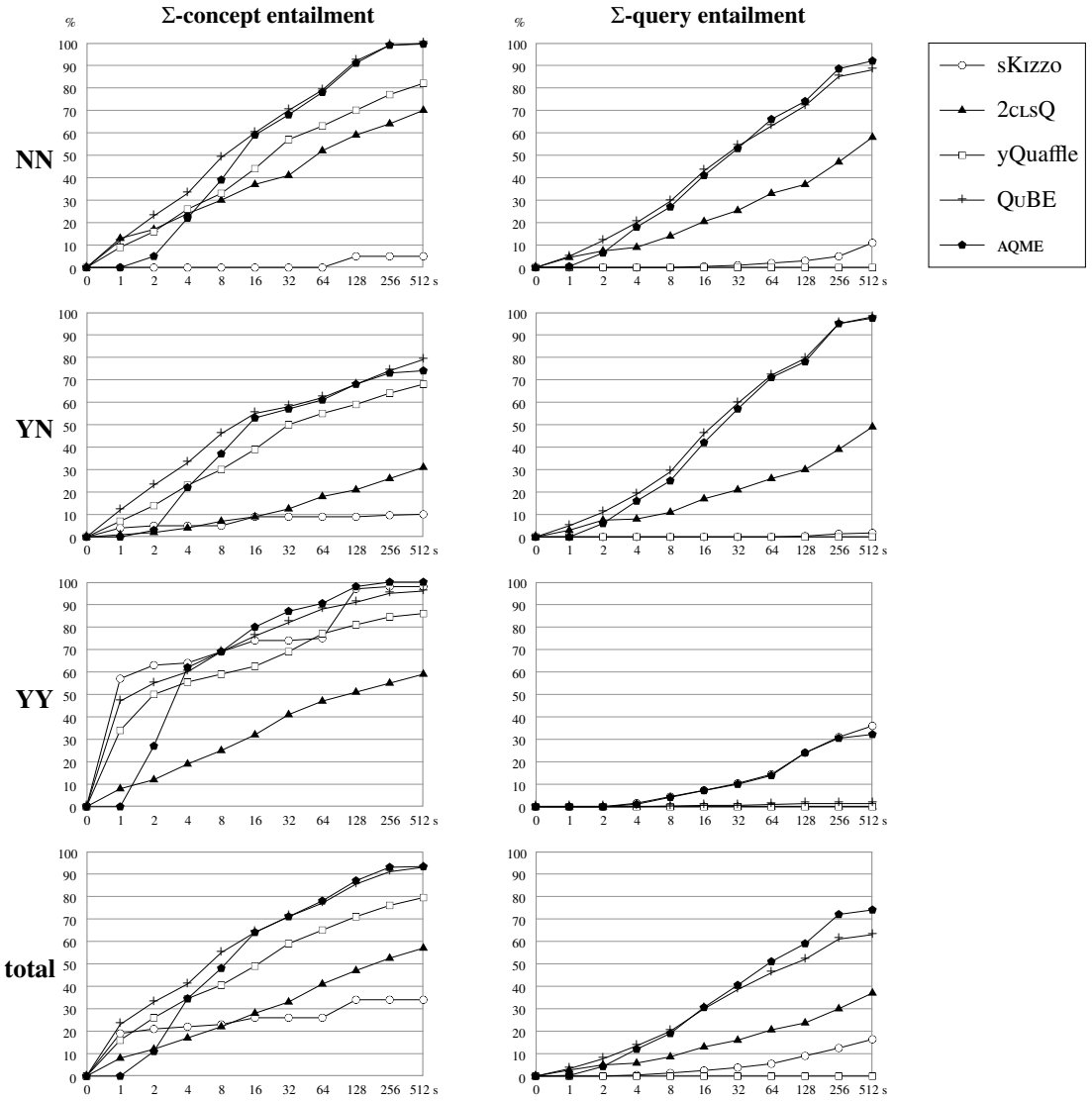

Figure 4: Percentage of solved instances.

Note the large difference between the sizes of the QBF translations for Σ-concept and Σ-query entailment (say, 74,621 v. 1,217,151 clauses in the same instance), which reflects the difference between QBFs (3) and (4) discussed above. Although of the same worst-case complexity, in practice Σ-concept entailment turns out to be much easier to check than Σ-query entailment; see Fig. 4, where the graphs in the left (right) column show the percentage of solved instances for Σ-concept (respectively, Σ-query) entailment.

We experimented with four standard QBF solvers: Skolemisation-based sKizzo [56] and search-based 2clsQ [57], yQuaffle [58, 59] and QuBE [60]. The tests were conducted on a 3GHz P4 machine with 2GB RAM.

It turned out that *none* of the four solvers was better than the others on all instances: for example, QuBE performed much stronger than sKizzo on the NN and YN series, but was outperformed by sKizzo on the (much harder) YY series. Moreover, *none* of the solvers could cope single-handedly with all of the tests and, even when the solver was successful, the runtime was quite unpredictable and could range from a few seconds to a few hours.

To select 'the best' QBF solver for each given instance, we employed the self-adaptive multi-engine system aqme [28], a tool capable of learning and choosing a QBF engine with 'more chances' to solve a given input. An important property of aqme is that it can update its learned policies when the usage scenario changes substantially by using an adaptation schema called *retraining*. Prior to the experiments, aqme computed a selection of syntactic

features (characterising the particular problems in question) from a pool of suitable QBF instances. A typical run of AQME is as follows. First, it leverages its inductive model (built using 1-nearest neighbour) to predict the best engine for a given input QBF. If the engine solves the QBF, AQME terminates and returns the answer. Otherwise, it starts its self-adaptive mechanism. It calls a different engine to solve the input formula. If it is successful, the retraining procedure is called and the inductive model is updated. Which engine is called for retraining and how much CPU time is granted to each engine are critical points for AQME's performance. As follows from Fig. 4, AQME indeed managed to select the best solver in all the cases, which was absolutely crucial for our module extraction experiments.

9.3. Practical module extraction

We extracted minimal modules from *DL-Lite*$^{\mathcal{N}}_{bool}$ encodings of two real-world commercial software applications called 'Core' and 'Umbrella'. The Core ontology is based on a supply-chain management system used by the book-store chain Ottakar's, now rebranded as Waterstone's. It contains 1283 axioms, 83 concept names and 77 role names, and features numerous functionality constraints, covering and disjointness constraints, and quite a few concepts of the form $\geq q\,R$ with $q > 2$. The Umbrella ontology is based on a specialised research data validation and processing system used by the Intensive Care National Audit and Research Centre (`http://www.icnarc.org`). It contains 1247 axioms, 79 concept names and 60 role names. Both ontologies are representations of the relevant data structures and were constructed by analysing the data model, database schema and application-level business logic. The 'publisher ontology' in Example 70 is part of Core.

We have conducted experiments with three types of minimal module extraction: for a *DL-Lite*$^{\mathcal{N}}_{bool}$ TBox $\mathcal{T}$ and a signature Σ, extract *some* minimal Σ-concept inseparability module (MCM) of $\mathcal{T}$, *some* minimal Σ-query inseparability module (MQM), and *the* minimal depleting Σ-query inseparability module (MDQM) of $\mathcal{T}$. As we have seen above, these extraction problems can be solved by the algorithms of Theorems 64 and 69 together with the 'QBF oracle' for deciding the Σ-concept and Σ-query inseparability relations. Unfortunately, a naïve implementation of this approach turns out to be hopelessly inefficient because to extract a minimal module from an ontology with, say, 1K axioms, even for 'typical' real-world examples the algorithm would call the oracle about 500K times. To reduce the numbers of such calls, we optimised the algorithms by checking a group of axioms rather than a single axiom at a time (in practice, this reduced the number of calls to 1–3K for a 1K ontology). On the other hand, to reduce the size of the original ontology, we 'pre-processed' it by means of the *tractable* syntactic locality-based algorithm from [17] extracting the so-called $\top\bot$*-module* ($\top\bot$M), which is a (not necessarily minimal) module with respect to the Σ-model inseparability relation, and so contains all the minimal modules we are interested in. In fact, we have the following inclusions:

$$\text{MCM} \subseteq \text{MQM} \subseteq \text{MDQM} \subseteq \top\bot\text{M},$$

where the first $\subseteq$ should be read as '*every* MQM contains *some* MCM', the second as '*every* MQM is contained in *the* MDQM', and the third $\subseteq$ as '*the* MDQM is contained in *the* $\top\bot$M'. Thus we can use these inclusions by computing modules from right to left.

Modules for $|\Sigma| = 1$. Our first experiment was to extract the modules of all three types, for all *singleton* signatures, from the full Core and Umbrella ontologies and from the corresponding pre-computed $\top\bot$Ms. For instance, the extracted MCM, MQM, and MDQM of Core for $\Sigma = \{\mathsf{Publisher}\}$ are given in Example 70 and contain 0, 3, and 20 axioms, respectively, whereas the corresponding $\top\bot$M has 228 axioms.

Table 2 summarises the results of the experiments (on average per module) in terms of module sizes. It also contains the average sizes of the segments extracted using the approaches described in [15] (SR), [14] (Prompt), and [16] (E-conn). Since these approaches do not support role names in the initial signature, we have only extracted modules for concept names in these cases. Furthermore, SR and Prompt are not logic-based and do not, in general, preserve entailments. (The Publisher-segments for SR, Prompt, and E-conn contain 19, 189, and 349 axioms, respectively.)

Table 3 compares the average extraction time and distribution of QBF engines calls for extracting MDQMs from the $\top\bot$M and the full ontology. The distribution of calls to the QBF engines changes notably if MDQMs are extracted from the whole ontology rather than from the $\top\bot$Ms: in the former case, the majority of calls is issued to sKIZZO, while QUBE handles most of the calls in the latter case. This complies with the observation that, in general, QUBE tends to solve easy instances more quickly, and sKIZZO performs more successfully on harder instances.

	Core	Umbrella
ontology	1283	1247
$\top\bot$M	226	69
MDQM	80	57
MQM	5	5
MCM	2	2
SR-segment	37	14
Prompt segment	162	102
E-conn module	243	47

Table 2: Ontology and average module sizes for singleton signatures.

	Core		Umbrella	
	$\top\bot$M	full	$\top\bot$M	full
extraction time	126s	2233s	60s	2488s
total AQME calls	385	565	254	463
sKIZZO	14%	76%	4%	74%
2CLSQ	2%	17%	1%	14%
QUBE	84%	7%	95%	12%

Table 3: Extraction time and distribution of calls for MDQM extracted from the $\top\bot$M and the full ontology.

Modules for $|\Sigma| = 10$. Then, for each of our ontologies, we randomly generated 30 signatures of 10 concept names each and extracted all possible modules; their average sizes are shown in Fig. 5. MCMs and MQMs were extracted from MDQMs, which in turn were extracted from $\top\bot$Ms. Again, in most of the AQME calls (1302/1694 for MDQMs and 152/181 for MCMs, on average) QUBE was invoked. The average runtime for MDQMs (MCMs) was around 30 (1.5) minutes.

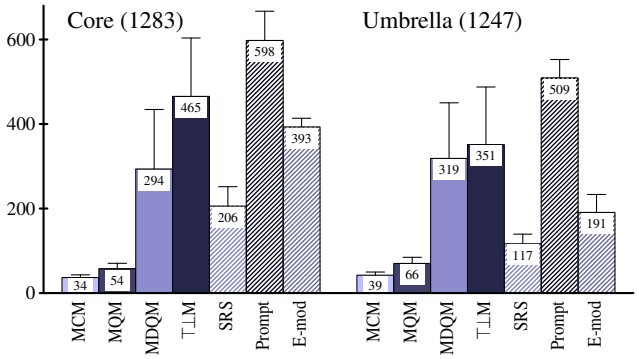

Figure 5: Module sizes for $|\Sigma| = 10$ and standard deviation.

It is to be noted that we have only been able to extract 15 MQMs for Umbrella and 5 MQMs for Core because the runtime for certain instances increases to a couple of days. One of the reasons is the growth of the QBF instances generated whenever the algorithm needs to test inseparability between module candidates and the original ontology. In the case of MDQMs, a candidate's complement needs to be compared with the empty TBox, which can be done rather efficiently. The case of MCMs and MQMs involves many comparisons of two very similar TBoxes, which, for MQMs, leads to the generation of QBF instances that are quadratic in the number of roles involved (as opposed to linear for MCMs; see Section 9.1).

10. Conclusion

We have introduced and analysed a framework for signature-based notions of difference, entailment and inseparability between ontologies in the description logics *DL-Lite*$^{\mathcal{N}}_{bool}$ and *DL-Lite*$^{\mathcal{N}}_{horn}$. These notions can be used to compare two versions of an ontology, to check whether importing one ontology into another has (possibly unwanted) side-effects, and to study and define refinements of a given ontology. We have also demonstrated that Σ-inseparability can be used as a framework for both module extraction and forgetting. Finally, we have presented promising experimental results of using QBF solvers to decide Σ-inseparability and extract (minimal) modules.

Many open problems remain. For example, it should be investigated how the Σ-difference between ontologies can be approximated in practice in such a way that the approximation provides the developers and users of ontologies with sufficient information to decide how different versions of ontologies can be reconciled and whether the differences are relevant for a certain application. Another open problem is to establish the precise computational complexity of Σ-model entailment and understand its practical applicability. We conjecture that, for *DL-Lite*$^{\mathcal{N}}_{bool}$ TBoxes corresponding to UML class diagrams or ER models used in practice, typically rather small counterexamples to Σ-model entailment exist and that discovering and computing them is often feasible. Finally, forgetting and uniform interpolation has not been developed for (strong) query-inseparability in *DL-Lite*$^{\mathcal{N}}_{horn}$, and experiments showing the size of uniform interpolants for real-world ontologies would be very desirable.

A. Appendix

Here we provide the omitted proofs of the statements from the previous sections. We begin by establishing a number of basic results that will be required in these proofs.

A.1. Preliminaries

The aim of this section is to introduce, in Lemma 81, an operation which allows us to amalgamate interpretations in a 'truth-preserving' way. We will need two simple definitions.

Given a signature Σ, we say that two interpretations $\mathcal{I}$ and $\mathcal{J}$ are Σ-*isomorphic* and write $\mathcal{I} \sim_\Sigma \mathcal{J}$ if there is a bijection $f\colon \Delta^{\mathcal{I}} \to \Delta^{\mathcal{J}}$ such that $f(a^{\mathcal{I}}) = a^{\mathcal{J}}$, for every object name a, $x \in A^{\mathcal{I}}$ if, and only if, $f(x) \in A^{\mathcal{J}}$, for every concept name A in Σ, and $(x, y) \in P^{\mathcal{I}}$ if, and only if, $(f(x), f(y)) \in P^{\mathcal{J}}$, for every role name P in Σ. Clearly, Σ-isomorphic interpretations cannot be distinguished by Σ-TBoxes, Σ-ABoxes or Σ-queries.

Given a family of interpretations $\mathcal{I}_i$, for $i \in \mathbf{I}$, with $0 \in \mathbf{I}$, define the interpretation

$$\mathcal{J} \;=\; \bigoplus_{i \in \mathbf{I}} \mathcal{I}_i,$$

where $\Delta^{\mathcal{J}} = \{(i, w) \mid i \in \mathbf{I}, w \in \Delta^{\mathcal{I}_i}\}$, $a^{\mathcal{J}} = (0, a^{\mathcal{I}_0})$, for an object name a, $A^{\mathcal{J}} = \{(i, w) \mid i \in \mathbf{I}, w \in A^{\mathcal{I}_i}\}$, for a concept name A, and $P^{\mathcal{J}} = \{((i, w_1), (i, w_2)) \mid i \in \mathbf{I}, (w_1, w_2) \in P^{\mathcal{I}_i}\}$, for a role name P. $\mathcal{J}$ will be called the *disjoint union* of the $\mathcal{I}_i$. The disjoint union of ω copies of an interpretation $\mathcal{I}$, that is, the disjoint union of the family $\mathcal{I}_i$, for $i \in \omega$ and $\mathcal{I}_i = \mathcal{I}$, will be denoted by $\mathcal{I}^\omega$:

$$\mathcal{I}^\omega \;=\; \bigoplus_{i \in \omega} \mathcal{I}_i.$$

It should be clear that Σ-TBoxes, Σ-ABoxes or Σ-queries (for *any* signature Σ) cannot distinguish between $\mathcal{I}$ and $\mathcal{I}^\omega$.

The following lemma provides an important model-theoretic property of *DL-Lite*$^{\mathcal{N}}_{bool}$ that will be frequently used to establish model-theoretic characterisations of various notions of Σ-entailment.

Lemma 81. *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be TBoxes in DL-Lite*$^{\mathcal{N}}_{bool}$*, Σ a signature, and let Ξ be both $\mathcal{T}_1$- and $\mathcal{T}_2$-precisely realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Then, for every at most countable model $\mathcal{I}_1$ of $\mathcal{T}_1$ precisely realising Ξ and every signature Σ' with $\Sigma' \cap sig(\mathcal{T}_2) \subseteq \Sigma$, there is a model $\mathcal{I}^*$ of $\mathcal{T}_2$ such that*

(i) $\mathcal{I}^* \sim_{\Sigma'} \mathcal{I}_1^\omega$;

(ii) $\mathcal{I}^*$ *precisely realises* Ξ;

In particular, if $sig(\mathcal{T}_1) \subseteq \Sigma'$ then $\mathcal{I}^*$ is a model of $\mathcal{T}_1 \cup \mathcal{T}_2$.

Proof. Let $\mathcal{I}_2$ be an at most countable model of $\mathcal{T}_2$ precisely realising Ξ. As both $\mathcal{I}_1^\omega$ and $\mathcal{I}_2^\omega$ realise each $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type from Ξ at countably infinitely many points, there is a bijection $f\colon \Delta^{\mathcal{I}_2^\omega} \to \Delta^{\mathcal{I}_1^\omega}$ which is *invariant under* $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Define $\mathcal{I}^*$ by taking $\Delta^{\mathcal{I}^*} = \Delta^{\mathcal{I}_2^\omega}$ and, for all object names a, concept names A and role names P,

$$a^{\mathcal{I}^*} = f^{-1}(a^{\mathcal{I}_1^\omega}), \quad A^{\mathcal{I}^*} = \begin{cases} \{x \mid f(x) \in A^{\mathcal{I}_1^\omega}\}, & \text{if } A \in \Sigma \cup \Sigma', \\ A^{\mathcal{I}_2^\omega}, & \text{otherwise,} \end{cases} \quad P^{\mathcal{I}^*} = \begin{cases} \{(x,y) \mid (f(x), f(y)) \in P^{\mathcal{I}_1^\omega}\}, & \text{if } P \in \Sigma \cup \Sigma', \\ P^{\mathcal{I}_2^\omega}, & \text{otherwise.} \end{cases}$$

By definition, $\mathcal{I}^* \sim_{\Sigma \cup \Sigma'} \mathcal{I}_1^\omega$. Therefore, $\mathcal{I}^* \sim_{\Sigma'} \mathcal{I}_1^\omega$ and $\mathcal{I}^*$ precisely realises Ξ.

Observe that each point x in $\Delta^{\mathcal{I}^*}$ has the same $sig(\mathcal{T}_2)Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type in $\mathcal{I}_2^\omega$ and $\mathcal{I}^*$. Indeed, as f is invariant under $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types, for a basic $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-concept B, we have $x \in B^{\mathcal{I}_2^\omega}$ if, and only if, $f(x) \in B^{\mathcal{I}_1^\omega}$ if, and only if, $x \in B^{\mathcal{I}^*}$. And, for a basic $(sig(\mathcal{T}_2) \setminus \Sigma)Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-concept B, $B^{\mathcal{I}^*} = B^{\mathcal{I}_2^\omega}$ by definition. Therefore, $\mathcal{I}^* \models \mathcal{T}_2$.

Finally, $sig(\mathcal{T}_1) \subseteq \Sigma'$ and $\mathcal{I}^* \sim_{\Sigma'} \mathcal{I}_1^\omega$ give $\mathcal{I}^* \models \mathcal{T}_1$. ❑

Next, we establish an analogue of Lemma 81 for TBoxes in *DL-Lite*$_{horn}^{\mathcal{N}}$ and sub-precise realisability. Given a signature Σ and interpretations $\mathcal{I}$ and $\mathcal{J}$, a map $f\colon \Delta^{\mathcal{I}} \to \Delta^{\mathcal{J}}$ is called a Σ-*homomorphism* if $a^{\mathcal{J}} = f(a^{\mathcal{I}})$, for every object name a, $x \in A^{\mathcal{I}}$ implies $f(x) \in A^{\mathcal{J}}$, for every concept name A in Σ and $x \in \Delta^{\mathcal{I}}$, and $(x,y) \in P^{\mathcal{I}}$ implies $(f(x), f(y)) \in P^{\mathcal{J}}$, for every role name P in Σ and $x, y \in \Delta^{\mathcal{I}}$. Queries in *DL-Lite*$_{horn}^{\mathcal{N}}$ are positive existential formulas and therefore, if there is a Σ-homomorphism from $\mathcal{I}$ to $\mathcal{J}$ then $\mathcal{I} \models \mathsf{q}(\boldsymbol{a})$ implies $\mathcal{J} \models \mathsf{q}(\boldsymbol{a})$, for every Σ-query $\mathsf{q}(\boldsymbol{x})$ in *DL-Lite*$_{horn}^{\mathcal{N}}$ and every tuple $\boldsymbol{a}$.

Lemma 82. *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be TBoxes in DL-Lite*$_{horn}^{\mathcal{N}}$*, Σ a signature, and let Ξ be a set of precisely $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types that is also sub-precisely $\mathcal{T}_2$-realisable. Then, for every countable model $\mathcal{I}_1$ of $\mathcal{T}_1$ precisely realising Ξ and every signature Σ' with $\Sigma' \cap sig(\mathcal{T}_2) \subseteq \Sigma$, there exist a model $\mathcal{I}^*$ of $\mathcal{T}_2$ realising all the types from Ξ and a Σ'-homomorphism from $\mathcal{I}^*$ onto $\mathcal{I}_1$. In particular, $\mathcal{I}^*$ sub-precisely realises Ξ.*

Proof. Let $\mathcal{I}_2$ be a countable model of $\mathcal{T}_2$ sub-precisely realising Ξ. Without loss of generality we may assume that the interpretations of all symbols not in $sig(\mathcal{T}_2)$ are empty. We construct a sequence of pairs (Δ_i, h_i), $i \in \omega$, where $\Delta_i \subseteq \Delta^{\mathcal{I}_2^\omega}$ and $h_i\colon \Delta_i \to \Delta^{\mathcal{I}_1}$ is a Σ-homomorphism from the Δ_i part of $\mathcal{I}_2^\omega$ onto $\mathcal{I}_1$, such that

$$\Delta_i \subseteq \Delta_{i+1}, \text{ for all } i \in \omega \quad \text{and} \quad \bigcup_{i \in \omega} \Delta_i = \Delta^{\mathcal{I}_2^\omega}.$$

To start with, choose $\Delta_0 \subseteq \Delta^{\mathcal{I}_2^\omega}$ and a bijection $h_0\colon \Delta_0 \to \Delta^{\mathcal{I}_1}$ in such a way that h_0 is invariant under $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types and each $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type realised in $\mathcal{I}_2^\omega$ is realised by countably infinitely many points in $\Delta^{\mathcal{I}_2^\omega} \setminus \Delta_0$. Such a bijection exists because $\mathcal{I}_2^\omega$ realises every $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type from Ξ countably infinitely many times.

Assume that an ordering $<$ of $\Delta^{\mathcal{I}_2^\omega}$ is isomorphic to ω, and suppose that (Δ_k, h_k) have already been constructed. To construct (Δ_{k+1}, h_{k+1}), we apply one of the following two rules to $x \in \Delta^{\mathcal{I}_2^\omega}$, provided that neither is applicable to any $y \in \Delta^{\mathcal{I}_2^\omega}$ with $y < x$.

- If $x \in \Delta_k$ and the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type of x in $\mathcal{I}_2^\omega$ contains $\geq q\, R$ such that $R^{\mathcal{I}_2^\omega} \cap (\Delta_k \times \Delta_k)$ has fewer than q pairs (x, x_i), pick a point $y \in \Delta^{\mathcal{I}_2^\omega} \setminus \Delta_k$ that has the same $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type as a point $z \in \Delta^{\mathcal{I}_1}$ with $(h_k(x), z) \in R^{\mathcal{I}_1}$ (this can be done since the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type of x in $\mathcal{I}_2^\omega$ is positively contained in the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type of $h_k(x)$ in $\mathcal{I}_1^\omega$). Then we set $\Delta_{k+1} = \Delta_k \cup \{y\}$ and $h_{k+1} = h_k \cup \{(y, z)\}$.

- If $x \in \Delta^{\mathcal{I}_2^\omega} \setminus \Delta_k$, select $z \in \Delta^{\mathcal{I}_1}$ such that the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type of x in $\mathcal{I}_2^\omega$ is positively contained in the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type of z in $\mathcal{I}_1$. Set $\Delta_{k+1} = \Delta_k \cup \{x\}$ and $h_{k+1} = h_k \cup \{(x, z)\}$.

Let $h = \bigcup_{i \in \omega} h_i$. Define an interpretation $\mathcal{I}^*$ by taking $\Delta^{\mathcal{I}^*} = \Delta^{\mathcal{I}_2^\omega}$ and, for all object names a, concept names A and role names P,

$$a^{\mathcal{I}^*} = h_0^{-1}(a^{\mathcal{I}_1}), \quad A^{\mathcal{I}^*} = \begin{cases} \{x \mid f(x) \in A^{\mathcal{I}_1^\omega}\}, & \text{if } A \in \Sigma \cup \Sigma', \\ A^{\mathcal{I}_2^\omega}, & \text{otherwise,} \end{cases} \quad P^{\mathcal{I}^*} = \begin{cases} \{(x,y) \mid (h(x), h(y)) \in P^{\mathcal{I}_1}\}, & \text{if } P \in \Sigma \cup \Sigma', \\ P^{\mathcal{I}_2^\omega}, & \text{otherwise.} \end{cases}$$

Clearly, the function h is a $\Sigma \cup \Sigma'$-homomorphism from $\mathcal{I}^*$ onto $\mathcal{I}_1$ (recall that in $\mathcal{I}_2$ the interpretations of symbols from $\Sigma' \setminus sig(\mathcal{T}_2)$ are empty) and the $sig(\mathcal{T}_2)Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type of each x is the same in $\mathcal{I}^*$ and $\mathcal{I}_2^\omega$. Hence $\mathcal{I}^*$ is a model of $\mathcal{T}_2$. ❑

We are now in a position to apply Lemma 81 and prove Theorem 73. Let Σ be a signature and Q a finite set of numerical parameters.

Theorem 73. *Let $\mathcal{L} \in \{\text{DL-Lite}^{\mathcal{N}}_{bool}, \text{DL-Lite}^{\mathcal{N}}_{horn}\}$. Then $\mathcal{L}$ has uniform interpolation, and a uniform interpolant of a TBox $\mathcal{T}$ with respect to Σ in $\mathcal{L}$ can be constructed effectively.*

Proof. Let $\mathcal{T}$ be a TBox in $\text{DL-Lite}^{\mathcal{N}}_{bool}$ and Σ a signature. Let $\mathcal{T}_\Sigma$ be the set of all concept inclusions $\bigsqcap_{C \in \boldsymbol{t}} C \sqsubseteq \bot$ such that $\boldsymbol{t}$ is a $\Sigma Q_{\mathcal{T}}$-type which is not $\mathcal{T}$-realisable. We show that $\mathcal{T}_\Sigma$ is a uniform interpolant for $\mathcal{T}$ with respect to Σ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$. Clearly $\mathcal{T}_\Sigma \models C \sqsubseteq D$ implies $\mathcal{T} \models C \sqsubseteq D$ for all concept inclusions $C \sqsubseteq D$. Conversely, assume that $\mathcal{T}_\Sigma \not\models C \sqsubseteq D$ and $sig(C \sqsubseteq D) \subseteq \Sigma'$, for a signature Σ' with $\Sigma' \cap sig(\mathcal{T}) \subseteq \Sigma$. Let $\mathcal{I}_1$ be a model of $\mathcal{T}_\Sigma$ such that $\mathcal{I}_1 \not\models C \sqsubseteq D$. We may assume that $\mathcal{I}$ is at most countable and realises the set Ξ of all $\mathcal{T}_\Sigma$-realisable $\Sigma Q_{\mathcal{T}}$-types. By the definition of $\mathcal{T}_\Sigma$, Ξ coincides with the set of all $\mathcal{T}$-realisable $\Sigma Q_{\mathcal{T}}$-types. Thus, Ξ is both $\mathcal{T}$ and $\mathcal{T}_\Sigma$-precisely realisable. So, by Lemma 81, we have a model $\mathcal{I}^* \sim_{\Sigma'} \mathcal{I}_1^\omega$ of $\mathcal{T}$. It follows that $\mathcal{I}^* \not\models C \sqsubseteq D$. Thus $\mathcal{T} \not\models C \sqsubseteq D$, as required.

Assume now that $\mathcal{T}$ is a $\text{DL-Lite}^{\mathcal{N}}_{horn}$ TBox. We define $\mathcal{T}'_\Sigma$ as the set of all concept inclusions $\bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B_0$ that follow from $\mathcal{T}$, where $\boldsymbol{t}$ is a $\Sigma Q_{\mathcal{T}}$-type that is not $\mathcal{T}$-realisable and $\neg B_0 \in \boldsymbol{t} \setminus \boldsymbol{t}^+$. It follows that $\mathcal{T}'_\Sigma$ implies $\mathcal{T}_\Sigma$. Indeed, $\mathcal{T}_\Sigma$ consists of concept inclusions of the form $\bigsqcap_{C \in \boldsymbol{t}} C \sqsubseteq \bot$, or, equivalently, $\bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq \bigsqcup_{\neg B \in \boldsymbol{t} \setminus \boldsymbol{t}^+} B$. As! Horn KBs enjoy the *disjunction property*, this means that there is $\neg B_0 \in \boldsymbol{t} \setminus \boldsymbol{t}^+$ such that $\mathcal{T}'_\Sigma \models \bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B_0$. So $\boldsymbol{t}$ is not $\mathcal{T}$-realisable and $\bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B_0 \in \mathcal{T}'_\Sigma$. Thus, $\mathcal{T}'_\Sigma$ is as required. ❑ polished

A.2. Proofs of results from Section 4

Now we use the technique developed in the previous section in order to prove the claims made in Section 4. Throughout this section we use the fact that if a type is realised then it is realised in an at most countable model (and similarly, if a set of types is precisely realisable then it is precisely realisable in an at most countable model).

Theorem 20. *The following conditions are equivalent for TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$ and a signature Σ:*

(ce_b) *$\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$;*

(r) *every $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type is $\mathcal{T}_2$-realisable.*

Proof. **(ce_b) $\Rightarrow$ (r)** Suppose that $\boldsymbol{t}$ is a $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type which is not $\mathcal{T}_2$-realisable. Then $\mathcal{T}_2 \models \bigsqcap_{C \in \boldsymbol{t}} C \sqsubseteq \bot$ but $\mathcal{T}_1 \not\models \bigsqcap_{C \in \boldsymbol{t}} C \sqsubseteq \bot$, contrary to $\mathcal{T}_2$ being Σ-concept entailed by $\mathcal{T}_1$.

(r) $\Rightarrow$ (ce_b) Suppose otherwise. Then there is $C_1 \sqsubseteq C_2$ with $sig(C_1 \sqsubseteq C_2) \subseteq \Sigma$ such that $\mathcal{T}_2 \models C_1 \sqsubseteq C_2$ and $\mathcal{T}_1 \not\models C_1 \sqsubseteq C_2$. Take the uniform interpolant $\mathcal{T}_{2\Sigma}$ of $\mathcal{T}_2$ with respect to Σ provided by Theorem 73. As $\mathcal{T}_2$ and $\mathcal{T}_{2\Sigma}$ are Σ-concept inseparable, we can find $C'_1 \sqsubseteq C'_2 \in \mathcal{T}_{2\Sigma}$ such that $\mathcal{T}_1 \not\models C'_1 \sqsubseteq C'_2$. And as $C'_1 \sqsubseteq C'_2$ is a $\Sigma Q_{\mathcal{T}_2}$-concept inclusion in $\text{DL-Lite}^{\mathcal{N}}_{bool}$, we can find a $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type that is not $\mathcal{T}_2$-realisable. Indeed, let $\mathcal{I}$ be a model of $\mathcal{T}_1$ with a point x such that $x \in (C'_1 \sqcap \neg C'_2)^{\mathcal{I}}$. Then the $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type realised at x in $\mathcal{I}$ is not $\mathcal{T}_2$-realisable. ❑

Theorem 21. *The following conditions are equivalent for TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$ and a signature Σ:*

(sce_b) *$\mathcal{T}_1$ strongly Σ-concept entails $\mathcal{T}_2$ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$;*

(qe_b) *$\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$;*

(sqe_b) *$\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ in $\text{DL-Lite}^{\mathcal{N}}_{bool}$;*

(pr) *every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is precisely $\mathcal{T}_2$-realisable.*

Proof. The implications **(sqe$_\mathbf{b}$)** $\Rightarrow$ **(qe$_\mathbf{b}$)** and **(sqe$_\mathbf{b}$)** $\Rightarrow$ **(sce$_\mathbf{b}$)** follow immediately from definitions.

(pr) $\Rightarrow$ **(sqe$_\mathbf{b}$)** Suppose there are a Σ-TBox $\mathcal{T}$, Σ-ABox $\mathcal{A}$ and Σ-query $\mathsf{q}(\boldsymbol{a})$ in *DL-Lite*$^{\mathcal{N}}_{bool}$ with $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ and $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$. Take a model $\mathcal{I}_1$ of $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A})$ such that $\mathcal{I}_1 \not\models \mathsf{q}(\boldsymbol{a})$ and let Ξ be the set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types realised in $\mathcal{I}_1$. By **(pr)**, Ξ is precisely $\mathcal{T}_2$-realisable. Then, by Lemma 81, there exists a model $\mathcal{I}^*$ of $\mathcal{T}_2$ such that $\mathcal{I}^* \sim_\Sigma \mathcal{I}_1^\omega$, and so $\mathcal{I}^* \models (\mathcal{T}, \mathcal{A})$ and $\mathcal{I}^* \not\models \mathsf{q}(\boldsymbol{a})$, contrary to $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$.

(qe$_\mathbf{b}$) $\Rightarrow$ **(pr)** Let Ξ be the set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types realised in a model $\mathcal{I}$ of $\mathcal{T}_1$, and let $\mathcal{A}_\Xi = \{C(a_{\boldsymbol{t}}) \mid C \in \boldsymbol{t},\ \boldsymbol{t} \in \Xi\}$, where $a_{\boldsymbol{t}}$ is a fresh object name for each $\boldsymbol{t} \in \Xi$. It follows that $\mathcal{I} \models (\mathcal{T}_1, \mathcal{A}_\Xi)$. Suppose that Ξ is not precisely $\mathcal{T}_2$-realisable. Then two cases are possible:

1. If, for every model $\mathcal{I}'$ of $\mathcal{T}_2$, there is some $\boldsymbol{t} \in \Xi$ that is not realised in $\mathcal{I}'$, i.e., $\mathcal{I}' \not\models \mathcal{A}_\Xi$, then consider the query $\mathsf{q} = \bot$: we have $(\mathcal{T}_2, \mathcal{A}_\Xi) \models \mathsf{q}$ but $(\mathcal{T}_1, \mathcal{A}_\Xi) \not\models \mathsf{q}$, which is a contradiction.
2. Otherwise, every model of $\mathcal{T}_2$ must realise a $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type that is not in Ξ. Let Θ be the set of all $\mathcal{T}_2$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types that are not in Ξ. As $\Theta \neq \emptyset$, we can take $\mathsf{q} = \exists x \bigvee_{\boldsymbol{t} \in \Theta} \bigwedge_{C \in \boldsymbol{t}} C(x)$. Then we have $(\mathcal{T}_2, \mathcal{A}_\Xi) \models \mathsf{q}$ but $(\mathcal{T}_1, \mathcal{A}_\Xi) \not\models \mathsf{q}$, which is again a contradiction.

(sce$_\mathbf{b}$) $\Rightarrow$ **(pr)** Let Ξ be a set of precisely $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types, and let $\mathcal{T}_\Xi = \{\top \sqsubseteq \bigsqcup_{\boldsymbol{t} \in \Xi} \bigsqcap_{C \in \boldsymbol{t}} C\}$. Then, for every $\boldsymbol{t} \in \Xi$, we have $\mathcal{T}_1 \cup \mathcal{T}_\Xi \not\models \bigsqcap_{C \in \boldsymbol{t}} C \sqsubseteq \bot$. Therefore, by **(sce$_\mathbf{b}$)**, $\mathcal{T}_2 \cup \mathcal{T}_\Xi \not\models \bigsqcap_{C \in \boldsymbol{t}} C \sqsubseteq \bot$, and thus there is a model $\mathcal{I}_{\boldsymbol{t}}$ of $\mathcal{T}_2 \cup \mathcal{T}_\Xi$ realising $\boldsymbol{t}$. Clearly, $\bigoplus_{\boldsymbol{t} \in \Xi} \mathcal{I}_{\boldsymbol{t}}$ is a model of $\mathcal{T}_2$ precisely realising the types in Ξ. ❑

Theorem 26. *For any TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{horn}$ and any signature Σ, the following conditions are equivalent*:

(qe$_\mathbf{h}$) *$\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{horn}$*;

(spr) *every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is sub-precisely $\mathcal{T}_2$-realisable.*

Proof. **(spr)** $\Rightarrow$ **(qe$_\mathbf{h}$)** Suppose there are a Σ-ABox $\mathcal{A}$ and a Σ-query $\mathsf{q}(\boldsymbol{a})$ in *DL-Lite*$^{\mathcal{N}}_{horn}$ such that $(\mathcal{T}_2, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ and $(\mathcal{T}_1, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$. Let $\mathcal{I}_1$ be a model of $(\mathcal{T}_1, \mathcal{A})$ with $\mathcal{I}_1 \not\models \mathsf{q}(\boldsymbol{a})$, and let Ξ be the set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types realised in $\mathcal{I}_1$. By **(spr)**, Ξ is sub-precisely $\mathcal{T}_2$-realisable. By Lemma 82, there is a model $\mathcal{I}^*$ of $\mathcal{T}_2$ and a Σ-homomorphism from $\mathcal{I}^*$ onto $\mathcal{I}_1$. But then $\mathcal{I}^* \models \mathcal{A}$ and $\mathcal{I}^* \not\models \mathsf{q}(\boldsymbol{a})$, from which $(\mathcal{T}_2, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$, contrary to our assumptions.

(qe$_\mathbf{h}$) $\Rightarrow$ **(spr)** Let Ξ be the set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types realised in a model $\mathcal{I}$ of $\mathcal{T}_1$, and let $\mathcal{A}_\Xi = \{B(a_{\boldsymbol{t}}) \mid B \in \boldsymbol{t}^+,\ \boldsymbol{t} \in \Xi\}$, where $a_{\boldsymbol{t}}$ is a fresh object name for each $\boldsymbol{t} \in \Xi$. It follows that $\mathcal{I} \models (\mathcal{T}_1, \mathcal{A}_\Xi)$. Suppose that Ξ is not sub-precisely $\mathcal{T}_2$-realisable. Then two cases are possible:

1. If, for every model $\mathcal{I}'$ of $\mathcal{T}_2$, there is $\boldsymbol{t} \in \Xi$ with $(\bigsqcap_{B \in \boldsymbol{t}^+} B)^{\mathcal{I}'} = \emptyset$, i.e., $\mathcal{I}' \not\models \mathcal{A}_\Xi$, then consider the query $\mathsf{q} = \bot$: we have $(\mathcal{T}_1, \mathcal{A}_\Xi) \not\models \mathsf{q}$ but $(\mathcal{T}_2, \mathcal{A}_\Xi) \models \mathsf{q}$, which is a contradiction.
2. Otherwise, every model $\mathcal{I}'$ of $\mathcal{T}_2$ satisfying $\mathcal{A}_\Xi$ must realise a $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type that is not positively contained in any type from Ξ. Let Θ be the set of all such $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. As $\Theta \neq \emptyset$, we can take $\mathsf{q} = \exists x \bigvee_{\boldsymbol{t} \in \Theta} \bigwedge_{B \in \boldsymbol{t}^+} B(x)$. Then $(\mathcal{T}_2, \mathcal{A}_\Xi) \models \mathsf{q}$ but $(\mathcal{T}_1, \mathcal{A}_\Xi) \not\models \mathsf{q}$, which is again a contradiction.

This completes the proof of the theorem. ❑

The following model-theoretic property of TBoxes in *DL-Lite*$^{\mathcal{N}}_{horn}$ is standard in Horn logic; see, e.g., [9].

Lemma 83. *Let $\mathcal{T}$ be a TBox in DL-Lite$^{\mathcal{N}}_{horn}$ and $\boldsymbol{t}$ a $\mathcal{T}$-realisable ΣQ-type. Then there exists a countable model $\mathcal{J}_\mathcal{T}(\boldsymbol{t})$ of $\mathcal{T}$ such that $\mathcal{J}_\mathcal{T}(\boldsymbol{t})$ realises $\boldsymbol{t}$ and, for every model $\mathcal{I}$ of $\mathcal{T}$ realising $\boldsymbol{t}$, there exists a Σ-homomorphism from $\mathcal{J}_\mathcal{T}(\boldsymbol{t})$ to $\mathcal{I}$.*

In what follows we fix some model $\mathcal{J}_\mathcal{T}(\boldsymbol{t})$ mentioned in the formulation of the lemma and call it the *minimal model of $\mathcal{T}$ realising $\boldsymbol{t}$*.

Given a realisable set Ξ of ΣQ-types, let the TBox $\mathcal{T}_\Xi$ contain all the ΣQ-concept inclusions

$$B_1 \sqcap \dots \sqcap B_k \sqsubseteq B$$

in *DL-Lite*$^{\mathcal{N}}_{horn}$ such that $B \in \boldsymbol{t}^+$ whenever $B_1, \dots, B_k \in \boldsymbol{t}^+$, for all $\boldsymbol{t} \in \Xi$. We will call $\mathcal{T}_\Xi$ the TBox *induced by* Ξ. Note that (i) if, for distinct ΣQ-concepts $B_1, \dots, B_k$, there is no $\boldsymbol{t} \in \Xi$ with $B_1, \dots, B_k \in \boldsymbol{t}^+$ then $B_1 \sqcap \dots \sqcap B_k \sqsubseteq \bot$ is in $\mathcal{T}_\Xi$, and (ii) if $B \in \boldsymbol{t}^+$, for all $\boldsymbol{t} \in \Xi$, then $\top \sqsubseteq B \in \mathcal{T}_\Xi$. The following lemma establishes a useful criterion for deciding meet-precise $\mathcal{T}$-realisability of a set Ξ in terms of $\mathcal{T} \cup \mathcal{T}_\Xi$-realisability:

Lemma 84. *Let Ξ be a set of ΣQ-types and $\boldsymbol{t}$ a ΣQ-type. Let $\Xi_{\boldsymbol{t}} = \{\boldsymbol{t}_i \in \Xi \mid \boldsymbol{t}^+ \subseteq \boldsymbol{t}_i^+\}$. Then $\boldsymbol{t}$ is $\mathcal{T}_\Xi$-realisable if, and only if, $\Xi_{\boldsymbol{t}} \neq \emptyset$ and $\boldsymbol{t}^+ = \bigcap_{\boldsymbol{t}_i \in \Xi_{\boldsymbol{t}}} \boldsymbol{t}_i^+$.*

In particular, Ξ is meet-precisely $\mathcal{T}$-realisable, for a TBox $\mathcal{T}$, if, and only if, every type in Ξ is $\mathcal{T} \cup \mathcal{T}_\Xi$-realisable.

Proof. ($\Rightarrow$) If $\Xi_{\boldsymbol{t}} = \emptyset$ then $\bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq \bot$ is in $\mathcal{T}_\Xi$, and so $\boldsymbol{t}$ cannot be $\mathcal{T}_\Xi$-realisable. If $\Xi_{\boldsymbol{t}} \neq \emptyset$ then, for every $B' \in \bigcap_{\boldsymbol{t}_i \in \Xi_{\boldsymbol{t}}} \boldsymbol{t}_i^+$, we have $\bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B'$ in $\mathcal{T}_\Xi$. So $\boldsymbol{t}^+ \supseteq \bigcap_{\boldsymbol{t}_i \in \Xi_{\boldsymbol{t}}} \boldsymbol{t}_i^+$.

($\Leftarrow$) If there is no model of $\mathcal{T}_\Xi$ realising $\boldsymbol{t}$, then $\mathcal{T}_\Xi \models \bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq \bigsqcup_{\neg B \in \boldsymbol{t} \setminus \boldsymbol{t}^+} B$, whence, by Theorem 24, there is $\neg B' \in \boldsymbol{t} \setminus \boldsymbol{t}^+$ such that $\mathcal{T}_\Xi \models \bigsqcap_{B \in \boldsymbol{t}^+} \sqsubseteq B'$, and so $\bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B'$ is in $\mathcal{T}_\Xi$. Therefore, $B' \in \boldsymbol{t}^+$, which is impossible. ❑

Lemma 85. *Let Ξ be a $\mathcal{T}$-realisable set of ΣQ-types. If a ΣQ-type $\boldsymbol{t}$ is $\mathcal{T}_\Xi$-realisable then $\boldsymbol{t}$ is $\mathcal{T}$-realisable.*

Proof. By Lemma 84, there are $\boldsymbol{t}_1, \dots, \boldsymbol{t}_k \in \Xi$ such that $\boldsymbol{t}^+ = \bigcap_i \boldsymbol{t}_i^+$. As the $\boldsymbol{t}_i$ are all realised in some model $\mathcal{I}$ of $\mathcal{T}$ and the intersection of models is a model for a Horn KB, $\boldsymbol{t}$ is $\mathcal{T}$-realisable. ❑

We are now in a position to prove the following criterion:

Theorem 27. *For any TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ in DL-Lite$_{horn}^{\mathcal{N}}$ and any signature Σ, the following conditions are equivalent*:

(sce$_\mathbf{h}$) *$\mathcal{T}_1$ strongly Σ-concept entails $\mathcal{T}_2$ in DL-Lite$_{horn}^{\mathcal{N}}$*;

(sqe$_\mathbf{h}$) *$\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ in DL-Lite$_{horn}^{\mathcal{N}}$*;

(mpr) *every precisely $\mathcal{T}_1$-realisable set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is meet-precisely $\mathcal{T}_2$-realisable*.

Proof. The implication **(sqe$_\mathbf{h}$)** $\Rightarrow$ **(sce$_\mathbf{h}$)** is trivial.

(mpr) $\Rightarrow$ **(sqe$_\mathbf{h}$)** Suppose there are a Σ-TBox $\mathcal{T}$, a Σ-ABox $\mathcal{A}$ and a Σ-query $\mathsf{q}(\boldsymbol{a})$ in *DL-Lite*$_{horn}^{\mathcal{N}}$ such that $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A}) \models \mathsf{q}(\boldsymbol{a})$ but $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$. Let $\mathcal{I}$ be a model of $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A})$ with $\mathcal{I} \not\models \mathsf{q}(\boldsymbol{a})$, and let Ξ be the set of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types realised in $\mathcal{I}$. Since $\mathcal{I} \models \mathcal{T}_\Xi$, every type in Ξ is $\mathcal{T}_\Xi$-realisable. Let Ξ^* be the set of all $\mathcal{T}_\Xi$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Consider

$$\mathcal{J} \quad = \quad \mathcal{I} \quad \oplus \quad \bigoplus_{\boldsymbol{t} \in \Xi^*} \mathcal{J}_{\mathcal{T}_\Xi}(\boldsymbol{t}).$$

As Ξ is $\mathcal{T}_1 \cup \mathcal{T}$-realisable, by Lemma 85, every $\mathcal{T}_\Xi$-realisable type is $\mathcal{T}_1 \cup \mathcal{T}$-realisable, and so $\mathcal{J} \models \mathcal{T}_1 \cup \mathcal{T}$. Clearly, we have $\mathcal{J} \models \mathcal{A}$ and, as there is a Σ-homomorphism from $\mathcal{J}$ onto $\mathcal{I}$, $\mathcal{J} \not\models \mathsf{q}(\boldsymbol{a})$. Also, observe that $\mathcal{J}$ precisely realises Ξ^*: indeed, $\mathcal{J}$ realises every $\mathcal{T}_\Xi$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type and, conversely, every $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type realised in $\mathcal{J}$ is $\mathcal{T}_\Xi$-realisable. By **(mpr)** and Lemma 84, there exists a model $\mathcal{I}'$ of $\mathcal{T}_2$ realising all the types in Ξ^* and such that each $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type realised in $\mathcal{I}'$ is $\mathcal{T}_{\Xi^*}$-realisable. In fact, $\mathcal{I}'$ realises precisely the set Ξ^*: since Ξ^* is $\mathcal{T}_\Xi$-realisable, by Lemma 85, every $\mathcal{T}_{\Xi^*}$-realisable type $\boldsymbol{t}$ is $\mathcal{T}_\Xi$-realisable, and so $\boldsymbol{t} \in \Xi^*$. We then apply Lemma 81 to $\mathcal{J}$ and Ξ^* and find a model $\mathcal{I}^*$ of $\mathcal{T}_2$ such that $\mathcal{I}^* \sim_\Sigma \mathcal{J}^\omega$. It follows that $\mathcal{I}^*$ is a model of $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A})$ such that $\mathcal{I}^* \not\models \mathsf{q}(\boldsymbol{a})$, which is a contradiction.

(sce$_\mathbf{h}$) $\Rightarrow$ **(mpr)** Let Ξ be a set of precisely $\mathcal{T}_1$-realisable $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Then $\mathcal{T}_1 \cup \mathcal{T}_\Xi \not\models \bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq \bigsqcup_{\neg B \in \boldsymbol{t} \setminus \boldsymbol{t}^+} B$, for each $\boldsymbol{t} \in \Xi$. Therefore, for each $\boldsymbol{t} \in \Xi$ and each $\neg B' \in \boldsymbol{t} \setminus \boldsymbol{t}^+$, we have $\mathcal{T}_1 \cup \mathcal{T}_\Xi \not\models \bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B'$, whence, by **(sce$_\mathbf{h}$)**, $\mathcal{T}_2 \cup \mathcal{T}_\Xi \not\models \bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq B'$. As an intersection of models of a Horn KB is also a model of this KB, we obtain $\mathcal{T}_2 \cup \mathcal{T}_\Xi \not\models \bigsqcap_{B \in \boldsymbol{t}^+} B \sqsubseteq \bigsqcup_{\neg B \in \boldsymbol{t} \setminus \boldsymbol{t}^+} B$, and thus $\boldsymbol{t}$ is $\mathcal{T}_2 \cup \mathcal{T}_\Xi$-realisable, for each $\boldsymbol{t} \in \Xi$. Take the disjoint union $\mathcal{J}$ of all models $\mathcal{I}_{\boldsymbol{t}}$ of $\mathcal{T}_2 \cup \mathcal{T}_\Xi$ realising $\boldsymbol{t}$, for $\boldsymbol{t} \in \Xi$. Clearly, $\mathcal{J}$ realises all the types in Ξ and each $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type realised in $\mathcal{J}$ is $\mathcal{T}_2 \cup \mathcal{T}_\Xi$-realisable. Therefore, by Lemma 84, Ξ is meet-precisely $\mathcal{T}_2$-realisable. ❑

A.3. Proofs of results from Section 5

Theorem 32. *All the entailment relations from Section 3 are robust under vocabulary extensions in DL-Lite$^{\mathcal{N}}_{bool}$ and DL-Lite$^{\mathcal{N}}_{horn}$.*

Proof. We go through the different notions of Σ-entailment.

(a) Σ-*concept entailment in DL-Lite$^{\mathcal{N}}_{bool}$.* This case follows from uniform interpolation of *DL-Lite$^{\mathcal{N}}_{bool}$*, as proved in Theorem 73 above. Suppose that $\mathcal{T}_1$ Σ-concept entails $\mathcal{T}_2$, $sig(\mathcal{T}_2) \cap \Sigma' \subseteq \Sigma$, $\mathcal{T}_2 \models C_1 \sqsubseteq C_2$, and $sig(C_1 \sqsubseteq C_2) \subseteq \Sigma'$. We have to show that $\mathcal{T}_1 \models C_1 \sqsubseteq C_2$. Let $\mathcal{T}_{2,\Sigma}$ be a uniform interpolant of $\mathcal{T}_2$ with respect to Σ in *DL-Lite$^{\mathcal{N}}_{bool}$*. Then $\mathcal{T}_1 \models \mathcal{T}_{2.\Sigma}$ and $\mathcal{T}_{2,\Sigma} \models C_1 \sqsubseteq C_2$, by the definition of uniform interpolants. Hence $\mathcal{T}_1 \models C_1 \sqsubseteq C_2$, as required.

(b) Σ-*concept entailment in DL-Lite$^{\mathcal{N}}_{horn}$.* This case follows from uniform interpolation of *DL-Lite$^{\mathcal{N}}_{horn}$* in the same way as in (a).

(c) Σ-*query entailment in DL-Lite$^{\mathcal{N}}_{bool}$* (and, equivalently, strong Σ-concept entailment and strong Σ-query entailment). Suppose that $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ and Σ' is a signature with $sig(\mathcal{T}_2) \cap \Sigma' \subseteq \Sigma$. We use the criterion of Theorem 21. Assume that there is a model $\mathcal{I}_1$ of $\mathcal{T}_1$ precisely realising a set Ξ of $\Sigma' Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Let $\Xi \restriction_\Sigma = \{\boldsymbol{t} \restriction_\Sigma \mid \boldsymbol{t} \in \Xi\}$. As $\Xi \restriction_\Sigma$ is $\mathcal{T}_1$-precisely realisable, it is also precisely $\mathcal{T}_2$-realisable. By Lemma 81, we then obtain a model $\mathcal{I}^*$ of $\mathcal{T}_2$ such that $\mathcal{I}^* \sim_{\Sigma'} \mathcal{I}_1^\omega$. Therefore, $\mathcal{I}^*$ precisely realises Ξ.

(d) Σ-*query entailment in DL-Lite$^{\mathcal{N}}_{horn}$.* Suppose $\mathcal{T}_1$ Σ-query entails $\mathcal{T}_2$ and Σ' is a signature with $sig(\mathcal{T}_2) \cap \Sigma' \subseteq \Sigma$. We use the criterion of Theorem 26. Assume that there is a model of $\mathcal{T}_1$ precisely realising a set Ξ of $\Sigma' Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. Let $\Xi \restriction_\Sigma = \{\boldsymbol{t} \restriction_\Sigma \mid \boldsymbol{t} \in \Xi\}$. As $\Xi \restriction_\Sigma$ is $\mathcal{T}_1$-precisely realisable, it is also sub-precisely $\mathcal{T}_2$-realisable. By Lemma 82, we then obtain a model $\mathcal{I}^*$ of $\mathcal{T}_2$ realising all the types in Ξ and a Σ'-homomorphism from $\mathcal{I}^*$ onto $\mathcal{I}_1$. Therefore, $\mathcal{I}^*$ sub-precisely realises Ξ.

(e) *Strong* Σ-*query entailment in DL-Lite$^{\mathcal{N}}_{horn}$* (and, equivalently, strong Σ-concept entailment). Suppose that $\mathcal{T}_1$ strongly Σ-concept entails $\mathcal{T}_2$ and Σ' is a signature with $sig(\mathcal{T}_2) \cap \Sigma' \subseteq \Sigma$. We use the criterion of Theorem 27. Assume that a set Ξ of $\Sigma' Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types is precisely $\mathcal{T}_1$-realisable. Consider the set Ξ^* of all $\mathcal{T}_\Xi$-realisable $\Sigma' Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types (constructed in the proof of Theorem 27, **(mpr)** $\Rightarrow$ **(sce$_{\mathbf{h}}$)**). It is precisely realised in some model $\mathcal{J}$. Let $\Xi^* \restriction_\Sigma = \{\boldsymbol{t} \restriction_\Sigma \mid \boldsymbol{t} \in \Xi^*\}$. As $\Xi^* \restriction_\Sigma$ is precisely $\mathcal{T}_1$-realisable (e.g., in $\mathcal{J}$), there exists a model $\mathcal{I}_2$ of $\mathcal{T}_2$ meet-precisely realising $\Xi^* \restriction_\Sigma$. By Lemma 84, every type in $\Xi^* \restriction_\Sigma$ is $\mathcal{T}_2 \cup \mathcal{T}_{\Xi^* \restriction_\Sigma}$-realisable, and by Lemma 85, $\mathcal{T}_{\Xi \restriction_\Sigma}$-realisable. Thus, $\mathcal{J}$ and $\mathcal{I}_2$ precisely realise the same set $\Xi \restriction_\Sigma$ of $\Sigma Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-types. By Lemma 81, we obtain a model $\mathcal{I}^*$ of $\mathcal{T}_2$ such that $\mathcal{I}^* \sim_{\Sigma'} \mathcal{J}^\omega$, and thus precisely realising Ξ^*. As $\Xi \subseteq \Xi^*$ and every $\Sigma' Q_{\mathcal{T}_1 \cup \mathcal{T}_2}$-type in Ξ is $\mathcal{T}_2 \cup \mathcal{T}_\Xi$-realisable, by Lemma 84, Ξ is meet-precisely $\mathcal{T}_2$-realisable.

(f) Σ-*model entailment.* Suppose that $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$ and Σ' is a signature with $sig(\mathcal{T}_2) \cap \Sigma' \subseteq \Sigma$. Let $\mathcal{I}$ be a model of $\mathcal{T}_1$. Then there exists a model $\mathcal{I}'$ of $\mathcal{T}_2$ that coincides with $\mathcal{I}$ on Σ. As $sig(\mathcal{T}_2) \cap \Sigma' \subseteq \Sigma$, we may actually assume that $\mathcal{I}'$ coincides with $\mathcal{I}$ of Σ'. Hence $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$. ❑

Theorem 36. *All the inseparability relations from Section 3 are robust under joins in DL-Lite$^{\mathcal{N}}_{bool}$ and DL-Lite$^{\mathcal{N}}_{horn}$.*

Proof. (a) Σ-*concept inseparability in DL-Lite$^{\mathcal{N}}_{bool}$.* Suppose that $\mathcal{T}$ and $\mathcal{T}_i$ are Σ-concept inseparable in *DL-Lite$^{\mathcal{N}}_{bool}$*, for $i = 1, 2$. Consider a $\mathcal{T}$-realisable ΣQ-type $\boldsymbol{t}$ with $Q = Q_{\mathcal{T} \cup \mathcal{T}_1 \cup \mathcal{T}_2}$. By Theorem 20, it is sufficient to show that $\boldsymbol{t}$ is $\mathcal{T}_1 \cup \mathcal{T}_2$-realisable. Let Ξ be the set of all $\mathcal{T}$-realisable ΣQ-types. As $\mathcal{T}$ and $\mathcal{T}_i$ are Σ-concept inseparable, Ξ is also the set of all $\mathcal{T}_i$-realisable ΣQ-types, for $i = 1, 2$. It follows that Ξ is precisely $\mathcal{T}_i$-realisable, for $i = 1, 2$. Using Lemma 81 with $\Sigma' = sig(\mathcal{T}_1)$, we obtain a model for $\mathcal{T}_1 \cup \mathcal{T}_2$ precisely realising Ξ. This model realises $\boldsymbol{t}$.

(b) Σ-*concept inseparability in DL-Lite$^{\mathcal{N}}_{horn}$.* This case follows from (a) by Theorem 24.

(c) Σ-*query inseparability in DL-Lite$^{\mathcal{N}}_{bool}$* (and, equivalently, strong Σ-concept inseparability and strong Σ-query inseparability). Suppose that $\mathcal{T}$ and $\mathcal{T}_i$ are Σ-query inseparable in *DL-Lite$^{\mathcal{N}}_{bool}$*, for $i = 1, 2$, and let Ξ be a precisely $\mathcal{T}$-realisable set of ΣQ-types, where $Q = Q_{\mathcal{T} \cup \mathcal{T}_1 \cup \mathcal{T}_2}$. By Theorem 21, it is sufficient to show that Ξ is precisely $\mathcal{T}_1 \cup \mathcal{T}_2$-realisable. By Theorem 21, Ξ is precisely $\mathcal{T}_i$-realisable, for $i = 1, 2$. Using Lemma 81, we obtain a model for $\mathcal{T}_1 \cup \mathcal{T}_2$ precisely realising Ξ.

(d) Σ-*query inseparability in DL-Lite$^{\mathcal{N}}_{horn}$.* Suppose that $\mathcal{T}$ and $\mathcal{T}_i$ are Σ-query inseparable in *DL-Lite$^{\mathcal{N}}_{horn}$*, for $i = 1, 2$, and let Ξ be a precisely $\mathcal{T}$-realisable set of ΣQ-types, where $Q = Q_{\mathcal{T} \cup \mathcal{T}_1 \cup \mathcal{T}_2}$. By Theorem 26, it is sufficient to

show that Ξ is sub-precisely $\mathcal{T}_1 \cup \mathcal{T}_2$-realisable. As $\mathcal{T}$ and $\mathcal{T}_i$ are Σ-query inseparable, we obtain a set $\Xi' \supseteq \Xi$ which is precisely $\mathcal{T}$-, $\mathcal{T}_1$-, and $\mathcal{T}_2$-realisable and such that each $\boldsymbol{t} \in \Xi'$ is positively contained in a type from Ξ. By Lemma 81, we obtain a model for $\mathcal{T}_1 \cup \mathcal{T}_2$ precisely realising Ξ'. But then Ξ is sub-precisely $\mathcal{T}_1 \cup \mathcal{T}_2$-realisable.

(e) *Strong Σ-query inseparability in DL-Lite$^{\mathcal{N}}_{horn}$* (and, equivalently, strong Σ-concept inseparability). Suppose that $\mathcal{T}$ and $\mathcal{T}_i$ are strongly Σ-query inseparable in DL-Lite$^{\mathcal{N}}_{horn}$, for $i = 1, 2$, and let Ξ be a precisely $\mathcal{T}$-realisable set of ΣQ-types, where $Q = Q_{\mathcal{T} \cup \mathcal{T}_1 \cup \mathcal{T}_2}$. Let $\mathcal{I}$ be a model of $\mathcal{T}$ precisely realising Ξ. Consider the set Ξ^* of all $\mathcal{T}_\Xi$-realisable ΣQ-types as in the proof of Theorem 27, **(mpr) $\Rightarrow$ (sce$_\mathbf{h}$)**. As follows from that proof, the set Ξ^* is precisely $\mathcal{T}_i$-realisable, for $i = 1, 2$. Hence, by Lemma 81, there exists a model for $\mathcal{T}_1 \cup \mathcal{T}_2$ precisely realising Ξ^*, and thus meet-precisely realising Ξ.

(e) *Σ-model inseparability*. Suppose that $\mathcal{T}$ and $\mathcal{T}_i$ are Σ-model inseparable and $sig(\mathcal{T}_1) \cap sig(\mathcal{T}_2) \subseteq \Sigma$. Let $\mathcal{I}$ be a model of $\mathcal{T}$. Then there are models $\mathcal{I}_1$ and $\mathcal{I}_2$ of $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively, that coincide with $\mathcal{I}$ on Σ. As $sig(\mathcal{T}_1) \cap sig(\mathcal{T}_2) \subseteq \Sigma$, we may assume that $\mathcal{I}_1 = \mathcal{I}_2$. Thus $\mathcal{I}_1$ is a model of $\mathcal{T}_1 \cup \mathcal{T}_2$ coinciding with $\mathcal{I}$ on Σ. ❑

Theorem 39. *If $\mathcal{T}_1$ Σ-query entails (or, equivalently, strongly Σ-concept entails) $\mathcal{T}_2$ in DL-Lite$^{\mathcal{N}}_{bool}$, then $\mathcal{T}_1$ strongly Σ-query entails $\mathcal{T}_2$ in $\mathcal{SHIQ}$.*

Proof. Suppose $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$. Take a model $\mathcal{J}$ of $(\mathcal{T}_1 \cup \mathcal{T}, \mathcal{A})$ such that $\mathcal{J} \not\models \mathsf{q}(\boldsymbol{a})$. By Lemma 81, we can find a model $\mathcal{I}^*$ of $\mathcal{T}_2$ such that $\mathcal{I}^* \sim_\Sigma \mathcal{J}^\omega$. But then $\mathcal{I}^*$ is a model of $(\mathcal{T} \cup \mathcal{T}_2, \mathcal{A})$ such that $\mathcal{I}^* \not\models \mathsf{q}(\boldsymbol{a})$. Hence $(\mathcal{T}_2 \cup \mathcal{T}, \mathcal{A}) \not\models \mathsf{q}(\boldsymbol{a})$. ❑

A.4. Proofs of results from Section 6

Lemma 50. *There is an algorithm which, given a TBox $\mathcal{T}$ in DL-Lite$^{\mathcal{N}}_{horn}$ and a set Ξ of ΣQ-types with $Q \supseteq Q_{\mathcal{T}}$, decides in deterministic polynomial time whether Ξ has a precise, sub-precise or meet-precise $\mathcal{T}$-witness and constructs such a witness if it exists.*

Proof. First we observe that, given a TBox $\mathcal{T}$ in DL-Lite$^{\mathcal{N}}_{horn}$ and a ΣQ-type $\boldsymbol{t}$ with $Q \supseteq Q_{\mathcal{T}}$, $\mathsf{cl}_{\mathcal{T}}(\boldsymbol{t})$ can be computed in polynomial time: just extend $\boldsymbol{t}$ with all the basic $sig(\mathcal{T})Q$-concepts B such that $B_1 \sqcap \dots \sqcap B_k \sqsubseteq B \in \mathcal{T}$ and the B_i are already in the computed extension of $\boldsymbol{t}$.

Let $\Xi = \{\boldsymbol{t}'_0, \dots, \boldsymbol{t}'_k\}$. In all three algorithms we first extend the types of Ξ to $sig(\mathcal{T})Q$-types and check whether they are $\mathcal{T}$-realisable (cf. $(\mathbf{w}_1)$ and $(\mathbf{w}_2)$ in Definition 45):

1. For each $0 \le i \le k$, compute $\boldsymbol{t}_i = \mathsf{cl}_{\mathcal{T}}(\boldsymbol{t}'_i)$ and check whether $\boldsymbol{t}_i \restriction_\Sigma = \boldsymbol{t}'_i$ and $\bot \notin \boldsymbol{t}_i$. If this is not the case for some i, stop with answer 'no' (see Proposition 23).

The types $\boldsymbol{t}_0, \dots, \boldsymbol{t}_k$ will form the first sequence of $sig(\mathcal{T})Q$-types in a $\mathcal{T}$-witness of Ξ (provided that it exists). So, it remains to construct the second sequence (and actually find all required witnesses for nonempty roles). The algorithm is iterative. To start with, we let $\boldsymbol{t}_j = \boldsymbol{t}_0$, for $k < j \le k + 2m$, where m is the number of role names in $\mathcal{T}$ (note that the choice of $\boldsymbol{t}_0$ is arbitrary). Also, let the set Ω_0 of 'processed' roles be empty.

Suppose we are at step n. Select a role name P_i from $\mathcal{T}$ that is nonempty and has not been processed yet (cf. $(\mathbf{w}_3)$), i.e., some $P_i \notin \Omega_n$ such that

$$\{\exists P_i, \exists P_i^-\} \cap \boldsymbol{t}_j \neq \emptyset, \quad \text{for} \quad 0 \le j \le k + 2m.$$

If no such role exists we terminate: $((\boldsymbol{t}_0, \dots, \boldsymbol{t}_k), (\boldsymbol{t}_{k+1}, \dots, \boldsymbol{t}_{m+2k}))$ is the required $\mathcal{T}$-witness for Ξ. Otherwise, compute $\boldsymbol{t}_{k+2i-1} = \mathsf{cl}_{\mathcal{T}}(\boldsymbol{t}_{\exists P_i})$ and $\boldsymbol{t}_{k+2i} = \mathsf{cl}_{\mathcal{T}}(\boldsymbol{t}_{\exists P_i^-})$, where $\boldsymbol{t}_B$ is the $sig(\mathcal{T})Q$-type such that $\boldsymbol{t}_B^+ = \{B\}$. Terminate with answer 'no' if either $\bot \in \boldsymbol{t}_{k+2i-1}$ or $\bot \in \boldsymbol{t}_{k+2i}$ (for these types are not $\mathcal{T}$-realisable, see Proposition 23 and $(\mathbf{w}_1)$). The next step of the algorithm depends on the particular type of witness: for $\boldsymbol{t}$ being both $\boldsymbol{t}_{k+2i-1}$ and $\boldsymbol{t}_{k+2i}$, we check

2-a. whether $\boldsymbol{t} \restriction_\Sigma = \boldsymbol{t}'$ for some $\boldsymbol{t}' \in \Xi$, if we need a precise $\mathcal{T}$-witness;

2-b. whether $\boldsymbol{t}^+ \restriction_\Sigma \subseteq \boldsymbol{t}'^+$ for some $\boldsymbol{t}' \in \Xi$, if we need a sub-precise $\mathcal{T}$-witness;

2-c. whether $\Xi_{\boldsymbol{t}} \neq \emptyset$ and $\boldsymbol{t}^+ \restriction_\Sigma = \bigcap_{\boldsymbol{t}_i \in \Xi_{\boldsymbol{t}}} \boldsymbol{t}_i'^+$, where $\Xi_{\boldsymbol{t}} = \{\boldsymbol{t}'_i \in \Xi \mid \boldsymbol{t}^+ \restriction_\Sigma \subseteq \boldsymbol{t}'_i\}$, if we need a meet-precise $\mathcal{T}$-witness.

Terminate with answer 'no' if the test fails. Otherwise, we update the types $\boldsymbol{t}_{k+2i-1}$ and $\boldsymbol{t}_{k+2i}$ of the sequence with the just computed ones and set $\Omega_{n+1} = \Omega_n \cup \{P_i\}$.

Clearly, the algorithm runs in polynomial time. ❑

Next, we provide proofs of Lemma 56 and Theorem 58. For Lemma 56, we have to construct a BAPA formula $\varphi_{P,q_{\max}}$ stating that a set-system $\mathcal{S}$ has a solution. For the construction we require, in addition to the notion of solution to $\mathcal{S}$, the following notion of left solution. Let

$$\mathcal{S} = (A_1, \ldots, A_{q_{\max}}, A_\infty), (B_1, \ldots, B_{q_{\max}}, B_\infty)$$

be a set-system. A relation ρ is called a *left solution* to $\mathcal{S}$ if

- A_q is the set of points of ρ-outdegree q, for $1 \leq q \leq q_{\max}$; A_∞ is the set of points of ρ-outdegree $> q_{\max}$;
- every point in B_q has ρ-indegree $\leq q$, for $1 \leq q \leq q_{\max}$; B_∞ is the set of points of ρ-indegree $> q_{\max}$.

Thus, the only difference between a left solution and a solution is that, in the former, points in B_q do not necessarily have ρ-indegree q, but can have ρ-indegree $\leq q$. First, we establish necessary and sufficient conditions for a set-system to have a (left) solution in some special case:

Lemma 86. *Let $q_{\max} < \omega$ and $\mathcal{S} = (A_1, \ldots, A_{q_{\max}}, \emptyset), (B_1, \ldots, B_{q_{\max}}, \emptyset)$ be a set-system.*

(B) *If $\mathcal{S}$ has a solution then*

$$\sum_{q=1}^{q_{\max}} q \cdot |A_q| \quad = \quad \sum_{q=1}^{q_{\max}} q \cdot |B_q|. \tag{5}$$

Conversely, if $\sum_{q=1}^{q_{\max}} |A_q| \geq q_{\max}^2$ or $\sum_{q=1}^{q_{\max}} |B_q| \geq q_{\max}^2$, then (5) implies that $\mathcal{S}$ has a solution.

(BL) *If $\mathcal{S}$ has a left solution then*

$$\sum_{q=1}^{q_{\max}} q \cdot |A_q| \quad \leq \quad \sum_{q=1}^{q_{\max}} q \cdot |B_q|. \tag{6}$$

Conversely, if $\sum_{q=1}^{q_{\max}} |A_q| \geq q_{\max}^2$ or $\sum_{q=1}^{q_{\max}} |B_q| \geq q_{\max}^2$, then (6) implies that $\mathcal{S}$ has a left solution.

Proof. **(B)**: It should be clear that (5) holds if $\mathcal{S}$ has a solution. Now suppose that (5) holds and $\sum_{q=1}^{q_{\max}} |A_q| \geq q_{\max}^2$. If the sums in (5) are infinite (i.e., at least one of the $|A_q|$ and one of the $|B_q|$ is an infinite cardinal), then a solution ρ is readily constructed. So we concentrate on the case where the sums are finite. Let $A = \bigcup_{q=1}^{q_{\max}} A_q$ and $B = \bigcup_{q=1}^{q_{\max}} B_q$. We first show that there exists a map

$$f \colon A \times B \to \mathbb{N}$$

such that, for each $1 \leq q \leq q_{\max}$,

$$\sum_{d \in B} f(a,d) = q, \quad \text{for every } a \in A_q, \qquad \text{and} \qquad \sum_{d \in A} f(d,b) = q, \quad \text{for every } b \in B_q. \tag{7}$$

Assume that such a map does not exist. Take a map $f \colon A \times B \to \mathbb{N}$ such that, for $1 \leq q \leq q_{\max}$,

$$\sum_{d \in B} f(a,d) \leq q, \quad \text{for every } a \in A_q, \qquad \text{and} \qquad \sum_{d \in A} f(d,b) \leq q, \quad \text{for every } b \in B_q \tag{8}$$

(the set of such maps is nonempty, e.g., $f(x,y) = 0$ satisfies (8)) and $\sum_{(x,y) \in A \times B} f(x,y)$ is maximal. By our assumption and (5),

$$\sum_{(x,y) \in A \times B} f(x,y) \quad < \quad \sum_{q=1}^{q_{\max}} q \cdot |A_q| \quad = \quad \sum_{q=1}^{q_{\max}} q \cdot |B_q|.$$

Thus, there exist q_1, q_2 and $a \in A_{q_1}$, $b \in B_{q_2}$ such that $\sum_{d\in B} f(a_1, d) < q_1$ and $\sum_{d\in A} f(d, b) < q_2$. Define f' by setting $f'(a, b) = f(a, b) + 1$ and $f'(x, y) = f(x, y)$ for all $(x, y) \in A \times B$ distinct from (a, b). Then (8) still holds for f' but $\sum_{(x,y)\in A\times B} f(x, y) < \sum_{(x,y)\in A\times B} f'(x, y)$ contrary to f having the maximal $\sum_{(x,y)\in A\times B} f(x, y)$.

We now show that there actually exists a map with (7) into $\{0, 1\}$. Suppose such a map does not exist. Take a map f with (7) such that $\sum_{f(x,y)>1} f(x, y) > 1$ is minimal and find a, b with $f(a, b) > 1$. We have $\sum_{q=1}^{q_{\max}} |A_q| \geq q_{\max}^2$, and so there exists $(a', b') \in A \times B$ such that $f(a', b') > 0$, $f(a, b') = 0$, and $f(a', b) = 0$. Indeed, let $C = \{c \mid f(a, c) > 0\}$ and $D = \{d \mid$ there is $c \in C$ with $f(d, c) > 0\}$. We have $|C| < q_{\max}$, and so $|D| < q_{\max}^2$. As $|A| \geq q_{\max}^2$, there exists $a' \in A \setminus D$ and, by (7), there exists $b' \in B$ with $f(a', b') > 0$. By construction, $f(a, b') = f(a', b) = 0$. Define a new map f_0 which coincides with f except that

$$f_0(a, b') = 1, \qquad f_0(a, b) = f(a, b) - 1, \qquad f_0(a', b) = 1, \qquad \text{and} \qquad f_0(a', b') = f(a', b') - 1.$$

Then f_0 still has (7) and $\sum_{f_0(x,y)>1} f_0(x, y) < \sum_{f(x,y)>1} f(x, y)$, contrary to $\sum_{f(x,y)>1} f(x, y)$ being minimal. The case when $\sum_{q=1}^{q_{\max}} |B_q| \geq q_{\max}^2$ is considered analogously.

Let $f\colon A \times B \to \{0, 1\}$ satisfy (7). Then the relation $\rho = \{(a, b) \in A \times B \mid f(a, b) = 1\}$ is a solution to $\mathcal{S}$.

(BL) is proved in the same way as **(B)** and is left to the reader. ❑

Note that the existence of a (left) solution to a set-system $\mathcal{S}$ does not depend on the sets themselves but only on their cardinalities. Thus, we can (and will) equivalently represent a set-system $\mathcal{S}$ in the form

$$\mathcal{S} = (n_1, \ldots, n_{q_{\max}}, n_\infty), (m_1, \ldots, m_{q_{\max}}, m_\infty),$$

where the n_i and m_i are cardinal numbers. In what follows, we will choose the representation most convenient for our purposes. The following lemma will be used to prove Lemma 56. It covers all four possible combinations and reduces the problem whether $\mathcal{S}$ has a solution to the special cases mentioned in Lemma 86.

Lemma 87. *Let $q_{\max} < \omega$. For a set-system $\mathcal{S} = (A_1, \ldots, A_{q_{\max}}, A_\infty), (B_1, \ldots, B_{q_{\max}}, B_\infty)$, one of the four cases holds*:

(C0) *If $|A_\infty| > q_{\max}$ and $|B_\infty| > q_{\max}$ then $\mathcal{S}$ has a solution.*

(C1) *If $|A_\infty| > q_{\max}$ and $|B_\infty| \leq q_{\max}$ then $\mathcal{S}$ has a solution if, and only if, the following holds*:

- *if $|B_\infty| = 0$ then $\mathcal{S}'_1 = (A_1, \ldots, A_{q_{\max}}, A_\infty, \emptyset), (B_1, \ldots, B_{q_{\max}}, \emptyset, \emptyset)$ has a left solution*;
- *if $|B_\infty| > 0$ then $\mathcal{S}'_2 = (A_{|B_\infty|+1}, \ldots, A_{q_{\max}}, A_\infty, \underbrace{\emptyset, \ldots, \emptyset}_{|B_\infty|}), (B_1, \ldots, B_{q_{\max}}, \emptyset)$ has a left solution.*

(C2) *If $|A_\infty| \leq q_{\max}$ and $|B_\infty| > q_{\max}$ then $\mathcal{S}$ has a solution if, and only if, the following holds*:

- *if $|A_\infty| = 0$ then $\mathcal{S}'_1 = (B_1, \ldots, B_{q_{\max}}, B_\infty, \emptyset), (A_1, \ldots, A_{q_{\max}}, \emptyset, \emptyset)$ has a left solution*;
- *if $|A_\infty| > 0$ then $\mathcal{S}'_2 = (B_{|A_\infty|+1}, \ldots, B_{q_{\max}}, B_\infty, \underbrace{\emptyset, \ldots, \emptyset}_{|A_\infty|}), (A_1, \ldots, A_{q_{\max}}, \emptyset)$ has a left solution.*

(C3) *If $|A_\infty| \leq q_{\max}$ and $|B_\infty| \leq q_{\max}$ then $\mathcal{S}$ has a solution if, and only if, there are numbers n_q^D, for $D \subseteq B_\infty$ and $1 \leq q \leq q_{\max}$, and m_q^D, for $D \subseteq A_\infty$ and $1 \leq q \leq q_{\max}$, such that the following holds*:

- $\sum\limits_{D\subseteq B_\infty} n_q^D = |A_q|$ *and* $\sum\limits_{D\subseteq A_\infty} m_q^D = |B_q|$, *for all* $1 \leq q \leq q_{\max}$;
- *for all* $e \in B_\infty$, $\sum\limits_{q=1}^{q_{\max}} \sum\limits_{\substack{D\subseteq B_\infty \\ e\in D}} n_q^D > q_{\max} - |A_\infty|$ *and, for all* $e \in A_\infty$, $\sum\limits_{q=1}^{q_{\max}} \sum\limits_{\substack{D\subseteq A_\infty \\ e\in D}} m_q^D > q_{\max} - |B_\infty|$;
- *and $\mathcal{S}' = (n'_1, \ldots, n'_{q_{\max}}, 0), (m'_1, \ldots, m'_{q_{\max}}, 0)$ has a solution, where*

$$n'_k = \sum_{q=1}^{q_{\max}} \sum_{\substack{D\subseteq B_\infty \\ q-|D|=k}} n_q^D \quad \text{and} \quad m'_k = \sum_{q=1}^{q_{\max}} \sum_{\substack{D\subseteq A_\infty \\ q-|D|=k}} m_q^D, \qquad \text{for } 1 \leq k \leq q_{\max}.$$

Proof. Let $A = \bigcup_{q=1}^{q_{\max}} A_q$ and $B = \bigcup_{q=1}^{q_{\max}} B_q$.

(C0): $|A_\infty| > q_{\max}$ and $|B_\infty| > q_{\max}$. For every $a \in A_q$, take a set $Y_a \subseteq B_\infty$ of cardinality q, and, for every $a \in A_\infty$, take a set $Y_a \subseteq B_\infty$ of cardinality $q_{\max}+1$. Such sets exist because $|B_\infty| > q_{\max}$. Similarly, for every $b \in B_q$, take a set $X_b \subseteq A_\infty$ of cardinality q, and for every $b \in B_\infty$, take a set $X_b \subseteq A_\infty$ of cardinality $q_{\max}+1$. Such sets exist because $|A_\infty| > q_{\max}$. Then the relation $\rho = \bigcup_{a\in A\cup A_\infty}(\{a\}\times Y_a) \cup \bigcup_{b\in B\cup B_\infty}(X_b \times \{b\})$ is clearly a solution to $\mathcal{S}$.

(C1): $|A_\infty| > q_{\max}$ and $|B_\infty| \le q_{\max}$. Consider first $|B_\infty| = 0$. Let ρ be a solution to $\mathcal{S}$. For every $a \in A_\infty$ such that $o_\rho(a) > q_{\max}+1$, we remove $(o_\rho(a) - q_{\max} - 1)$-many pairs (a,b) from ρ and denote the resulting binary relation by ρ'. Then $o_{\rho'}(a) = q_{\max}+1$ for all $a \in A_\infty$, and so ρ' is a left solution to $\mathcal{S}'_1$. Conversely, assume that $\mathcal{S}'_1$ has a left solution ρ. For any $q \le q_{\max}$ and $b \in B_q$ such that $i_\rho(b) < q$, take $(q - i_\rho(b))$-many points $a \in A_\infty$ such that $(a,b) \notin \rho$ and add the pairs (a,b) to ρ. This is possible because $|A_\infty| > q_{\max}$. Then the resulting relation ρ' is a solution to $\mathcal{S}$.

The claim for $|B_\infty| > 0$ is proved similarly and left to the reader.

(C2): $|A_\infty| \le q_{\max}$ and $|B_\infty| > q_{\max}$. This is a mirror image of **(C1)**.

(C3): $|A_\infty| \le q_{\max}$ and $|B_\infty| \le q_{\max}$. Suppose that $\mathcal{S}$ has a solution ρ. Define A^D_q, for $D \subseteq B_\infty$, $1 \le q \le q_{\max}$, and B^D_q, for $D \subseteq A_\infty$, $1 \le q \le q_{\max}$, by taking

$$\begin{aligned} A^D_q &= \{\, a \in A_q \mid \forall b \in B_\infty\ ((a,b) \in \rho \leftrightarrow b \in D)\}, \\ B^D_q &= \{\, b \in B_q \mid \forall a \in A_\infty\ ((a,b) \in \rho \leftrightarrow d \in D)\}. \end{aligned}$$

Thus, e.g., A^D_q are the points in A_q that are ρ-related to exactly the points in $D \subseteq B_\infty$. We show that the numbers $n^D_q = |A^D_q|$ and $m^D_q = |B^D_q|$ satisfy the (in)equalities of **(C3)** and that $\mathcal{S}'$ has a solution. The equality $\sum_{D\subseteq B_\infty} n^D_q = |A_q|$ follows from the fact that each $a \in A_q$ is in exactly one set of the form A^D_q. Let $e \in B_\infty$. Then $\sum_{q=1}^{q_{\max}} \sum_{e\in D\subseteq B_\infty} n^D_q$ is the number of points a with $(a,e) \in \rho$ such that $a \notin A_\infty$. This number must be greater than $(q_{\max} - |A_\infty|)$ because $i_\rho(a) > q_{\max}$ and there are at most $|A_\infty|$ points $a \in A_\infty$ with $(a,e) \in \rho$. The numbers m^D_q are considered in the same way. Consider now the restriction ρ' of ρ to $A \times B$. Then the number of points $a \in A$ with $o_\rho(a) = k$ is n'_k and the number of points $b \in B$ with $i_\rho(b) = k$ is m'_k. Thus, ρ' is a solution to $(n'_1, \dots, n'_{q_{\max}}, 0)$, $(m'_1, \dots, m'_{q_{\max}}, 0)$, as required.

For the converse direction, suppose that we have numbers n^q_D, m^q_D satisfying the conditions of **(C3)**. Let ρ be a solution to $(n'_1, \dots, n'_{q_{\max}}, 0)$,$(m'_1, \dots, m'_{q_{\max}}, 0)$. We may assume that ρ is a solution to a system $(A'_1, \dots, A'_{q_{\max}}, \emptyset)$, $(B'_1, \dots, B'_{q_{\max}}, \emptyset)$ in which

- each A'_k is the disjoint union of sets $\hat{A}^D_k \subseteq A$ of cardinality n^D_q, for $D \subseteq B_\infty$ and $q - |D| = k$;
- each B'_k is the disjoint union of sets $\hat{B}^D_k \subseteq B$ of cardinality m^D_q, for $D \subseteq A_\infty$ and $q - |D| = k$.

Now, for each $a \in \hat{A}^D_k$ and each $b \in \hat{B}^{D'}_k$, we add to ρ the pairs (a,d), $d \in D$, and the pairs (d',b), $d' \in D'$. Denote the resulting relation by ρ_0. It follows from $\sum_{D\subseteq B_\infty} n^D_q = |A_q|$ that the number of points a with $o_{\rho_0}(a) = q$ is $|A_q|$. Similarly, from $\sum_{D\subseteq A_\infty} m^D_q = |B_q|$ we obtain that the number of points b with $i_{\rho_0}(b) = q$ is $|B_q|$. For $e \in B_\infty$, using the inequality $\sum_{q=1}^{q_{\max}} \sum_{e\in D\subseteq B_\infty} n^D_q > q_{\max} - |A_\infty|$, we can expand ρ_0 by sufficiently many pairs (e', e) with $e' \in A_\infty$ so that the indegree of each $e \in B_\infty$ is at least $q_{\max}+1$. Similarly, for $e \in A_\infty$, using the inequality $\sum_{q=1}^{q_{\max}} \sum_{e\in D\subseteq A_\infty} m^D_q > q_{\max} - |B_\infty|$, we can expand ρ_0 by sufficiently many pairs (e, e') with $e' \in B_\infty$ so that the outdegree of each $e \in A_\infty$ is at least $q_{\max}+1$. The resulting relation ρ is a solution to the set-system $(A_1, \dots, A_{q_{\max}}, A_\infty)$, $(B_1, \dots, B_{q_{\max}}, B_\infty)$. ❑

We are now in a position to prove the following:

Lemma 56. *For every role name P and every number $q_{\max} \ge 1$, one can construct a BAPA formula $\varphi_{P,q_{\max}}$ with free variables*

$$X_{=1P}, \dots, X_{=q_{\max}P}, X_{>q_{\max}P},\ X_{=1P^-}, \dots, X_{=q_{\max}P^-}, X_{>q_{\max}P^-} \tag{9}$$

such that, for every BAPA model $\mathfrak{M}$, the following conditions are equivalent:

(i) $\mathfrak{M} \models \varphi_{P,q_{\max}}$;

(ii) *the set-system* $(X^{\mathfrak{M}}_{=1P}, \dots, X^{\mathfrak{M}}_{=q_{\max}P}, X^{\mathfrak{M}}_{>q_{\max}P}), (X^{\mathfrak{M}}_{=1P^-}, \dots, X^{\mathfrak{M}}_{=q_{\max}P^-}, X^{\mathfrak{M}}_{>q_{\max}P^-})$ *has a solution.*

Proof. The formula $\varphi_{P,q_{\max}}$ is defined by a case distinction similar to the formulation of Lemma 87 above. Namely, we define $\varphi_{P,q_{\max}}$ as the conjunction of the formulas:

- $(|X_{>q_{\max}P}| > q_{\max}) \wedge (|X_{>q_{\max}P^-}| > q_{\max}) \rightarrow (0 = 0)$,
- $(|X_{>q_{\max}P}| > q_{\max}) \wedge (|X_{>q_{\max}P^-}| \leq q_{\max}) \rightarrow \psi_1$,
- $(|X_{>q_{\max}P}| \leq q_{\max}) \wedge (|X_{>q_{\max}P^-}| > q_{\max}) \rightarrow \psi_2$,
- $(|X_{>q_{\max}P}| \leq q_{\max}) \wedge (|X_{>q_{\max}P^-}| \leq q_{\max}) \rightarrow \psi_3$.

The first conjunct corresponds to **(C0)** of Lemma 87 stating that a solution exists whenever the cardinalities of the sets of points of outdegree and, respectively, indegree $> q_{\max}$ are greater than $q_{\max}$. To define formulas ψ_1, ψ_2, ψ_3, note first that one can trivially construct a BAPA formula $\varphi^0_{P,q_{\max}}$ satisfying the conditions of Lemma 56 for all models $\mathfrak{M}$ with $\left(\sum_{q=1}^{q_{\max}} |X^{\mathfrak{M}}_{=q\,P}| < q^2_{\max}\right)$ and $\left(\sum_{q=1}^{q_{\max}} |(X_{=q\,P^-})^{\mathfrak{M}}| < q^2_{\max}\right)$ by simply listing all possible configurations of cardinalities $< q_{\max}$ of the free variables in (9) for which solutions exist. Now, according to **(B)** of Lemma 86, we can define a formula ψ^B with the intended meaning

$$\text{`}(X^{\mathfrak{M}}_{=1P}, \ldots, X^{\mathfrak{M}}_{=q_{\max}P}, \emptyset), (X^{\mathfrak{M}}_{=1P^-}, \ldots, X^{\mathfrak{M}}_{=q_{\max}P^-}, \emptyset) \text{ has a solution'}$$

by taking

$$\Big(\sum_{q=1}^{q_{\max}}(q \cdot |X_{=q\,P}|) \;=\; \sum_{q=1}^{q_{\max}}(q \cdot |X_{=q\,P^-}|)\Big) \quad \wedge \quad \Big((\sum_{q=1}^{q_{\max}}(|X_{=q\,P}| < q^2_{\max}) \wedge \sum_{q=1}^{q_{\max}}(|X_{=q\,P^-}| < q^2_{\max})) \rightarrow \varphi^0_{P,q_{\max}}\Big).$$

Similarly, by using the condition of **(BL)** in Lemma 86, we can define a BAPA formula ψ^{BL} stating that a set-system $(X^{\mathfrak{M}}_{=1P}, \ldots, X^{\mathfrak{M}}_{=q_{\max}P}, \emptyset), (X^{\mathfrak{M}}_{=1P^-}, \ldots, X^{\mathfrak{M}}_{=q_{\max}P^-}, \emptyset)$ has a left solution.

And by Lemma 87, ψ_1, ψ_2 and ψ_3 can be constructed from ψ^B and ψ^{BL} (with appropriate renaming of variables). We leave this rather tedious but straightforward construction to the interested reader. ❑

In the remainder of this section we give a proof of Theorem 58 stating that deciding Σ-model entailment for TBoxes in $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{horn}$ with maximal numerical parameter $q_{\max} = 3$ is coNExpTime-hard. The proof is by reduction of the model-conservativity problem for modal logic $\mathcal{S}5$. Recall that formulas of the *propositional modal language* $\mathcal{ML}$ are constructed from propositional variables $p_1, p_2, \ldots$ using the Booleans $\wedge$, $\neg$, and the modal (possibility) operator $\Diamond$. $\mathcal{ML}$-formulas are interpreted in (Kripke) models of the form $\mathcal{K} = (\Delta^{\mathcal{K}}, p_1^{\mathcal{K}}, p_2^{\mathcal{K}}, \ldots)$, where $\Delta^{\mathcal{K}}$ is a nonempty set and $p_i^{\mathcal{K}} \subseteq \Delta^{\mathcal{K}}$ for all propositional variables p_i. The interpretation $\varphi^{\mathcal{K}}$ of a modal formula φ in $\mathcal{K}$ is defined inductively as follows:

$$\begin{array}{rcl} (\psi_1 \wedge \psi_2)^{\mathcal{K}} & = & \psi_1^{\mathcal{K}} \cap \psi_2^{\mathcal{K}}, \\ (\neg\psi)^{\mathcal{K}} & = & \Delta^{\mathcal{K}} \setminus \psi^{\mathcal{K}}, \\ (\Diamond\psi)^{\mathcal{K}} & = & \{d \in \Delta^{\mathcal{K}} \mid \exists d' \in \psi^{\mathcal{K}}\}. \end{array}$$

A *global formula* is a modal formula in which every propositional variable is in the scope of a $\Diamond$. Observe that, for every global formula φ and every model $\mathcal{K}$, we have $\varphi^{\mathcal{K}} \in \{\emptyset, \Delta^{\mathcal{K}}\}$. We say that $\mathcal{K}$ is a *model* of a global formula φ (or that φ is *true* in $\mathcal{K}$) if $\varphi^{\mathcal{K}} = \Delta^{\mathcal{K}}$.

A global formula φ_2 is said to be a (*finite*) *model conservative extension* of a global formula φ_1 if, for every (finite) model $\mathcal{K}$ of φ_1, there exists a model $\mathcal{K}'$ of φ_2 such that $\Delta^{\mathcal{K}} = \Delta^{\mathcal{K}'}$ and $p_i^{\mathcal{K}} = p_i^{\mathcal{K}'}$, for all variables p_i of φ_1. The following result is proved in [53].[3]

[3] Note that this is result is not formulated explicitly in [53] but follows immediately from the proof of [53, Theorem 4] stating that the conservativeness problem is coNExpTime-hard for a large family of normal modal logics including $\mathcal{S}5$.

Theorem 88. (i) *For any global modal formulas φ_1 and φ_2, φ_2 is a model conservative extension of φ_1 if, and only if, φ_2 is a finite model conservative extension of φ_1.*

(ii) *It is* NExpTime*-hard to decide whether a global formula φ_2 is not a (finite) model-conservative extension of a global formula φ_1.*

We first present a reduction of model conservativity in $\mathcal{S}5$ to Σ-model entailment between *DL-Lite*$^{\mathcal{N}}_{bool}$ TBoxes. Then we modify this reduction to obtain a reduction of finite model-conservativity in $\mathcal{S}5$ to Σ-model entailment between *DL-Lite*$^{\mathcal{N}}_{horn}$ TBoxes.

Fix global modal formulas φ_1 and φ_2. Denote by $s(\varphi_i)$ the closure under single negation of the set of subformulas of φ_i. For every $\psi \in s(\varphi_i)$, take a concept name A_ψ and, additionally, for every $\Diamond\psi \in s(\varphi_i)$, take three role names $S_{\Diamond\psi}$, $L_{\Diamond\psi}$ and $S_{\neg\Diamond\psi}$. Let Dom and Box be fresh concept names.

The extensions of Dom will be employed to simulate the domains of $\mathcal{S}5$-models. Thus, the interpretation $\psi^{\mathcal{K}}$ of a subformula ψ of φ_i will correspond to $(A_\psi \sqcap \mathsf{Dom})^{\mathcal{I}}$ in the description logic interpretation $\mathcal{I}$. (We could work with $\mathsf{Dom}^{\mathcal{I}} = \Delta^{\mathcal{I}}$ as well but prefer allowing Dom to be a proper subset of $\Delta^{\mathcal{I}}$ because that will be necessary for the encoding into *DL-Lite*$^{\mathcal{N}}_{horn}$.)

We assemble a TBox $\mathcal{T}_1$ by first encoding the truth-conditions for $\wedge$ and $\neg$ in the obvious manner by taking

$$\neg A_\psi \sqcap \mathsf{Dom} \equiv A_{\neg\psi} \sqcap \mathsf{Dom}, \qquad \text{for all } \neg\psi \in s(\varphi_1), \tag{10}$$

$$A_{\psi_1} \sqcap A_{\psi_2} \sqcap \mathsf{Dom} \equiv A_{\psi_1\wedge\psi_2} \sqcap \mathsf{Dom}, \qquad \text{for all } \psi_1 \wedge \psi_2 \in s(\varphi_1). \tag{11}$$

To encode the truth condition for $\Diamond$ we use, besides $A_{\Diamond\psi}$, the role names $S_{\Diamond\psi}$, $S_{\neg\Diamond\psi}$ and $L_{\Diamond\psi}$. First we state that, for every $\Diamond\psi \in s(\varphi_1)$, the extensions of $\exists S_{\Diamond\psi}$ and $A_{\Diamond\psi}$ as well as their negations coincide on Dom:

$$\mathsf{Dom} \sqcap A_{\Diamond\psi} \equiv \mathsf{Dom} \sqcap \exists S_{\Diamond\psi}, \qquad \mathsf{Dom} \sqcap A_{\neg\Diamond\psi} \equiv \mathsf{Dom} \sqcap \exists S_{\neg\Diamond\psi}. \tag{12}$$

Next, we state that $S_{\Diamond\psi}$ and $S_{\neg\Diamond\psi}$, for $\Diamond\psi \in s(\varphi_1)$, are binary relations between Dom and Box:

$$\exists S_{\Diamond\psi} \sqsubseteq \mathsf{Dom}, \qquad \exists S^-_{\Diamond\psi} \sqsubseteq \mathsf{Box}, \qquad \exists S_{\neg\Diamond\psi} \sqsubseteq \mathsf{Dom}, \qquad \exists S^-_{\neg\Diamond\psi} \sqsubseteq \mathsf{Box}. \tag{13}$$

Finally, we connect A_ψ and $A_{\Diamond\psi}$ (via $S_{\Diamond\psi}$, $S_{\neg\Diamond\psi}$, and $L_{\Diamond\psi}$) by stating, for every $\Diamond\psi \in s(\varphi_1)$,

$$A_\psi \sqcap \mathsf{Dom} \sqsubseteq \exists S_{\Diamond\psi}, \qquad \exists S_{\Diamond\psi} \sqsubseteq \exists L_\psi, \qquad \exists L^-_\psi \sqsubseteq \mathsf{Dom} \sqcap A_\psi, \qquad \exists S^-_{\Diamond\psi} \sqcap \exists S^-_{\neg\Diamond\psi} \sqsubseteq \bot. \tag{14}$$

These inclusions together enforce that $A_{\Diamond\psi} \sqcap \mathsf{Dom}$ simulates $\Diamond\psi$ in models where Box is a *singleton set*: in such a model $\mathcal{I}$ we have $(\exists S_{\Diamond\psi})^{\mathcal{I}} = \emptyset$ or $(\exists S_{\neg\Diamond\psi})^{\mathcal{I}} = \emptyset$, by the last inclusion of (14) and the condition that the range of $S_{\Diamond\psi}$ and $S_{\neg\Diamond\psi}$ is a subset of Box. Using the remaining inclusions in (14), (12) and (13), we obtain that $(A_\psi \sqcap \mathsf{Dom})^{\mathcal{I}} \neq \emptyset$ if, and only if, $(\exists S_{\Diamond\psi})^{\mathcal{I}} \neq \emptyset$ if, and only if, $(\exists S_{\Diamond\psi})^{\mathcal{I}} = \mathsf{Dom}$ if, and only if, $(A_{\Diamond\psi} \sqcap \mathsf{Dom})^{\mathcal{I}} = \mathsf{Dom}^{\mathcal{I}}$, as required. Finally, to say that φ_1 is true we take:

$$\mathsf{Dom} \sqsubseteq A_{\varphi_1}. \tag{15}$$

Thus, $\mathcal{T}_1$ consists of the concept inclusions (10)–(15).

We construct $\mathcal{T}_2$ in the same way (but now taking concept inclusions for $\psi \in s(\varphi_2)$), except that we do no include $\mathsf{Dom} \sqsubseteq A_{\varphi_2}$ corresponding to (15) into $\mathcal{T}_2$. Instead, we take a set of inclusions that forces, when not satisfiable, Box to be a singleton set. To this end, we consider three fresh role names S_0, S, S' and include into $\mathcal{T}_2$ the following axioms stating that S and S' are functions from Dom to Box (we leave the inclusions for S' to the reader):

$$\exists S \equiv \mathsf{Dom}, \qquad \exists S^- \sqsubseteq \mathsf{Box}, \qquad (\geq 2\, S) \sqsubseteq \bot. \tag{16}$$

We also add an axiom saying that if a point is in the range of both S and S', then it is in the domain of S_0:

$$\exists S^- \sqcap \exists S'^- \sqsubseteq \exists S_0. \tag{17}$$

Finally, we encode that φ_2 is true by taking

$$\exists S_0^- \sqsubseteq \mathsf{Dom} \sqcap A_{\varphi_2}. \tag{18}$$

Lemma 89. *φ_2 is a model conservative extension of φ_1 if, and only if, $\mathcal{T}_1$ Σ-model entails $\mathcal{T}_2$, for $\Sigma = sig(\mathcal{T}_1)$.*

Proof. Suppose φ_2 is not a model conservative extension of φ_1. Take a model $\mathcal{K}$ of φ_1 for which there is no model $\mathcal{K}'$ of φ_2 having the same domain and the same interpretations of the variables in φ_1 as $\mathcal{K}$ (in this case, we simply say that $\mathcal{K}$ cannot be *expanded to a model of* φ_2). Define an interpretation $\mathcal{I}$ by taking $\Delta^{\mathcal{I}} = \Delta^{\mathcal{K}}$, $\mathsf{Dom}^{\mathcal{I}} = \Delta^{\mathcal{I}}$, $\mathsf{Box}^{\mathcal{I}} = \{d\}$, for some $d \in \Delta^{\mathcal{I}}$, and

(a) $A_\psi^{\mathcal{I}} = \psi^{\mathcal{K}}$, for all $\psi \in s(\varphi_1)$;

(b) $S^{\mathcal{I}}_{\Diamond\psi} = (\Diamond\psi)^{\mathcal{K}} \times \{d\}$ and $S^{\mathcal{I}}_{\neg\Diamond\psi} = (\neg\Diamond\psi)^{\mathcal{K}} \times \{d\}$, for all $\Diamond\psi \in s(\varphi_1)$;

(c) $L_\psi^{\mathcal{I}} = \mathsf{Dom}^{\mathcal{I}} \times \psi^{\mathcal{K}}$, for all $\Diamond\psi \in s(\varphi_1)$.

It is readily checked that $\mathcal{I}$ is a model of $\mathcal{T}_1$. We show that it cannot be expanded to a model of $\mathcal{T}_2$. Assume that such an expansion $\mathcal{I}'$ exists. Then, by (16)–(18) and $\mathsf{Box}^{\mathcal{I}}$ being a singleton set, $A_{\varphi_2}^{\mathcal{I}'}$ is nonempty. Define an expansion $\mathcal{K}'$ of $\mathcal{K}$ by setting $p^{\mathcal{K}'} = A_p^{\mathcal{I}'}$ for all those variables p in φ_2 that do not occur in φ_1. Using the fact that $\mathsf{Box}^{\mathcal{I}}$ is a singleton set, one can show by induction that $\psi^{\mathcal{K}'} = A_\psi^{\mathcal{I}'}$ for all $\psi \in s(\varphi_2)$. Hence $\varphi_2^{\mathcal{K}'}$ is nonempty (and so coincides with the domain of $\mathcal{K}'$), which is a contradiction.

Conversely, assume that $\mathcal{T}_1$ does not Σ-model entail $\mathcal{T}_2$. Let $\mathcal{I}$ be a witness model—i.e., a model of $\mathcal{T}_1$ that cannot be expanded to a model of $\mathcal{T}_2$. We first show that $\mathsf{Box}^{\mathcal{I}}$ is a singleton set. Assume that this is not the case. Choose distinct $d, d' \in \mathsf{Box}^{\mathcal{I}}$ and define an extension $\mathcal{I}_0$ of $\mathcal{I}$ by taking $S^{\mathcal{I}_0} = \mathsf{Dom}^{\mathcal{I}} \times \{d\}$, $S'^{\mathcal{I}_0} = \mathsf{Dom}^{\mathcal{I}} \times \{d'\}$, and $S_0^{\mathcal{I}_0} = \emptyset$. Then $\mathcal{I}_0$ is a model of (16)–(18) independently of the interpretation of A_{φ_2}. It is straightforward to interpret the remaining fresh symbols of $\mathcal{T}_2$ in such a way that $\mathcal{I}_0$ is a model of $\mathcal{T}_2$, contrary to our assumption.

Now take a model $\mathcal{K}$ with domain $\mathsf{Dom}^{\mathcal{I}}$ and $p^{\mathcal{K}} = A_p^{\mathcal{I}} \cap \mathsf{Dom}^{\mathcal{I}}$, for all variables p in φ_1. Using the fact that $\mathsf{Box}^{\mathcal{I}}$ is a singleton set, it is easily checked by induction that, for all $\psi \in s(\varphi_1)$,

$$\psi^{\mathcal{K}} = A_\psi^{\mathcal{I}} \cap \mathsf{Dom}^{\mathcal{I}}.$$

Thus, $\mathcal{K}$ is a model of φ_1. We show that there does not exist an expansion $\mathcal{K}'$ of $\mathcal{K}$ that is a model of φ_2. Assume such a $\mathcal{K}'$ exists. Define an expansion $\mathcal{I}'$ of $\mathcal{I}$ by setting $A_\psi^{\mathcal{I}'} = \psi^{\mathcal{K}'}$ for all new $\psi \in s(\varphi_2)$. Then, $A_{\varphi_2}^{\mathcal{I}'} \supseteq \mathsf{Dom}^{\mathcal{I}}$. Define extensions of $S_{\Diamond\psi}$, $S_{\neg\Diamond\psi}$ and L_ψ as in (b)–(c) above (now using $\mathcal{K}'$ for $\Diamond\psi \in s(\varphi_2)$). Let $S^{\mathcal{I}'}$ and $S'^{\mathcal{I}'}$ be functions from $\mathsf{Dom}^{\mathcal{I}}$ to $\mathsf{Box}^{\mathcal{I}}$ and let $S_0^{\mathcal{I}'}$ be a function from $\mathsf{Box}^{\mathcal{I}}$ to $\mathsf{Dom}^{\mathcal{I}}$. It is readily checked that $\mathcal{I}'$ is a model of $\mathcal{T}_2$, which is a contradiction. ❑

We now modify the encoding above with the aim of obtaining an encoding into *DL-Lite*$^{\mathcal{N}}_{horn}$. Observe that the only 'problematic' axiom is (10) encoding negation. To construct a *DL-Lite*$^{\mathcal{N}}_{horn}$ TBox $\mathcal{T}_1'$, we take (11)–(15) and replace (10) as follows. First, we state, using an auxiliary role name P, that the whole domain is exactly twice as large as Dom:

$$(\geq 3\, P) \sqsubseteq \bot, \quad \mathsf{Dom} \equiv (\geq 2\, P), \quad (\geq 2\, P^-) \sqsubseteq \bot, \quad \top \sqsubseteq \exists P^-. \tag{19}$$

We also state that A_ψ and $A_{\neg\psi}$ are disjoint: for $\psi \in s(\varphi_1)$,

$$A_\psi \sqcap A_{\neg\psi} \sqsubseteq \bot. \tag{20}$$

Finally, we add that Dom and A_ψ have the same cardinality for $\psi \in s(\varphi_1)$. To this end, we use a fresh role name P_ψ and say that P_ψ is a bijection from Dom onto A_ψ:

$$\mathsf{Dom} \equiv \exists P_\psi, \quad A_\psi \equiv \exists P_\psi^-, \quad (\geq 2\, P_\psi) \sqsubseteq \bot, \quad (\geq 2\, P_\psi^-) \sqsubseteq \bot. \tag{21}$$

The *DL-Lite*$^{\mathcal{N}}_{horn}$ TBox $\mathcal{T}_1'$ consists of concept inclusions (11)–(15) and (19)–(21).

Lemma 90. *If $\mathcal{I}$ is a finite model of $\mathcal{T}_1'$ then, for all $\psi \in s(\varphi_1)$,*

$$(A_{\neg\psi} \sqcap \mathsf{Dom})^{\mathcal{I}} = (\neg A_\psi \sqcap \mathsf{Dom})^{\mathcal{I}}.$$

Proof. By (20), $A^{\mathcal{I}}_{\psi}$ and $A^{\mathcal{I}}_{\neg\psi}$ are disjoint. Thus, it is sufficient to show that $A^{\mathcal{I}}_{\psi} \cup A^{\mathcal{I}}_{\neg\psi} \supseteq \mathsf{Dom}^{\mathcal{I}}$. In fact, as $A^{\mathcal{I}}_{\psi}$, $A^{\mathcal{I}}_{\neg\psi}$, $\mathsf{Dom}^{\mathcal{I}}$ and $\Delta^{\mathcal{I}} \setminus \mathsf{Dom}^{\mathcal{I}}$ all have the same cardinality and $\Delta^{\mathcal{I}}$ is finite, a simple pigeonhole argument shows that $A^{\mathcal{I}}_{\psi} \cup A^{\mathcal{I}}_{\neg\psi} = \Delta^{\mathcal{I}}$. ❑

Note that Lemma 90 does not hold for *infinite* models. To construct $\mathcal{T}'_2$, we take the concept inclusions from $\mathcal{T}'_1$ (formulated for $\psi \in s(\varphi_2)$) except (15). We add axioms (16) and (17) from the definition of $\mathcal{T}_2$. For $\mathcal{T}'_2$, however, it is not sufficient to add (18) as we do not only have to ensure that $\mathsf{Box}^{\mathcal{I}}$ is a singleton set but also that $\mathcal{I}$ is *finite*. To this end we replace (18) by the axioms saying that a fresh role name Z is an injective function from $\Delta^{\mathcal{I}}$ to $\Delta^{\mathcal{I}}$:

$$\top \sqsubseteq \exists Z, \qquad (\geq 2\, Z) \sqsubseteq \bot, \qquad (\geq 2\, Z^{-}) \sqsubseteq \bot \tag{22}$$

together with the following concept inclusion stating that if a point is in the range of Z and the range of S_0, then it is in $\mathsf{Dom} \sqcap A_{\varphi_2}$:

$$\exists S_0^{-} \sqcap \exists Z^{-} \sqsubseteq \mathsf{Dom} \sqcap A_{\varphi_2}. \tag{23}$$

To understand the purpose of these axioms, recall that a set $\Delta^{\mathcal{I}}$ is finite if, and only if, there does not exist an injective function from $\Delta^{\mathcal{I}}$ to $\Delta^{\mathcal{I}}$ that is not surjective. Thus, if an interpretation $\mathcal{I}$ is infinite, then we can always expand it to a model $\mathcal{I}'$ of (22) and (23) by choosing an injective but non-surjective function Z whose range is disjoint from the range of S_0 (as we can always choose an S_0 having only one point in its range). On the other hand, if $\mathcal{I}$ is finite, then (22) and (23) enforce, in the same way as (18) above, that $\mathsf{Dom} \sqcap A_{\varphi_2}$ is nonempty if Box is a singleton set.

The following lemma can be proved using this observation and combining the proof of Lemma 89 with Lemma 90:

Lemma 91. *φ_2 is a finite model conservative extension of φ_1 if, and only if, $\mathcal{T}'_1$ Σ-model entails $\mathcal{T}'_2$, where $\Sigma = sig(\mathcal{T}'_1)$.*

Remark 92. It is worth mentioning that the TBox $\mathcal{T}'_1$ constructed above can be used to show that finite model reasoning in *DL-Lite*$^{\mathcal{N}}_{horn}$ is *non-tractable* (in contrast to the results of [61] showing that, in other logics of the *DL-Lite* family, finite model reasoning is *tractable*). Indeed, consider the *DL-Lite*$^{\mathcal{N}}_{horn}$ TBox $\mathcal{T}'_1$ for a *propositional* formula φ_1—i.e., assume that the modal operator $\Diamond$ does not occur in φ_1. Then φ_1 is satisfiable if, and only if, the concept $A_{\varphi_1} \sqcap \mathsf{Dom}$ is satisfiable in a finite model of $\mathcal{T}'_1$. Thus, the problem whether a *DL-Lite*$^{\mathcal{N}}_{horn}$ concept is satisfiable in finite model of a *DL-Lite*$^{\mathcal{N}}_{horn}$ TBox is NP-hard.

A.5. Proofs of results from Section 8

Here we prove Theorem 78 and Theorem 79.

Theorem 78. *Let $\mathcal{T}$ and $\mathcal{T}'$ be TBoxes in DL-Lite*$^{\mathcal{N}}_{bool}$ *and Σ a signature. And let $\mathcal{T}'_\Sigma$ be a uniform query interpolant of $\mathcal{T}'$ with respect to Σ in DL-Lite*$^{u}_{bool}$*. Then $\mathcal{T}$ Σ-query entails $\mathcal{T}'$ if, and only if, $\mathcal{T} \models C_1 \sqsubseteq C_2$, for every $(C_1 \sqsubseteq C_2) \in \mathcal{T}'_\Sigma$.*

Proof. Suppose that $\mathcal{T}$ Σ-query entails $\mathcal{T}'$ and $\mathcal{T} \not\models \varkappa$, for some $\varkappa \in \mathcal{T}'_\Sigma$. Let $\mathcal{I}$ be a model of $\mathcal{T}$ such that $\mathcal{I} \not\models \varkappa$. Let Q be the set of numerical parameters in $\mathcal{T} \cup \mathcal{T}' \cup \{\varkappa\}$ and Ξ the set of ΣQ-types realised in $\mathcal{I}$. Then Ξ is $\mathcal{T}$-precisely realisable. Hence, by Theorem 21, Ξ is $\mathcal{T}'$-precisely realisable. Let $\mathcal{I}'$ be a model of $\mathcal{T}'$ precisely realising Ξ. Then $\mathcal{I}' \not\models \varkappa$ because $\mathcal{I}$ and $\mathcal{I}'$ realise the same ΣQ-types. It follows that $\mathcal{T}' \not\models \varkappa$, and so $\varkappa \notin \mathcal{T}'_\Sigma$, which is a contradiction.

Conversely, suppose that $\mathcal{T}$ does not Σ-query entail $\mathcal{T}'$. By Theorem 21, there exists a set Ξ of $\Sigma Q_{\mathcal{T}\cup\mathcal{T}'}$-types which is precisely $\mathcal{T}$-realisable but not precisely $\mathcal{T}'$-realisable. Let

$$D \;=\; \Big(\bigsqcap_{t\in\Xi} \exists \mathbf{U}.\bigsqcap_{C\in t} C\Big) \;\sqcap\; \Big(\forall \mathbf{U}.\bigsqcup_{t\in\Xi}\bigsqcap_{C\in t} C\Big),$$

where $\forall \mathbf{U}.C' = \neg\exists \mathbf{U}.\neg C'$. Then $\mathcal{T} \not\models D \sqsubseteq \bot$ but $\mathcal{T}' \models D \sqsubseteq \bot$. It follows that $\mathcal{T}'_\Sigma \models D \sqsubseteq \bot$. So there exists $\varkappa \in \mathcal{T}'_\Sigma$ such that $\mathcal{T} \not\models \varkappa$. ❑

Theorem 79. *For every TBox $\mathcal{T}$ in DL-Lite*$^{\mathcal{N}}_{bool}$ *and every signature Σ, one can construct a uniform query interpolant $\mathcal{T}_\Sigma$ of $\mathcal{T}$ with respect to Σ in DL-Lite*$^{u}_{bool}$*.*

Proof. Let $\mathcal{T}$ be a TBox in *DL-Lite*$^{\mathcal{N}}_{bool}$ and Σ a signature. Let m be the number of role names in $\mathcal{T}$. Define $\mathcal{T}_\Sigma$ to be the set containing all concept inclusions of the form $\bigsqcap_{C\in\boldsymbol{t}} C \sqsubseteq \bot$, where $\boldsymbol{t}$ is a $\Sigma Q_\mathcal{T}$-type which is not $\mathcal{T}$-realisable, as well as all concept inclusions of the form

$$\bigsqcap_{C\in\boldsymbol{t}} C \;\sqsubseteq\; \bigsqcup_{\Xi\in\boldsymbol{\Omega}} \Big(\bigsqcap_{\boldsymbol{t}'\in\Xi} \exists \mathbf{U}.\bigsqcap_{C\in\boldsymbol{t}'} C\Big),$$

where $\boldsymbol{t}$ is a $\mathcal{T}$-realisable $\Sigma Q_\mathcal{T}$-type and $\boldsymbol{\Omega}$ is the set of all sets Ξ of $\Sigma Q_\mathcal{T}$-types with $|\Xi| \leq 2m+1$ such that $\{\boldsymbol{t}\}\cup\Xi$ is precisely $\mathcal{T}$-realisable. It follows that $\mathcal{T}_\Sigma$ can be constructed in exponential time in the size of $\mathcal{T}$. It remains to show that $\mathcal{T}_\Sigma$ is a uniform query interpolant. Clearly, $\mathcal{T} \models \varkappa$, for all $\varkappa \in \mathcal{T}_\Sigma$. For the converse direction, it is sufficient to show that each precisely $\mathcal{T}_\Sigma$-realisable set of $\Sigma Q_\mathcal{T}$-types is precisely $\mathcal{T}$-realisable. Let Ξ_0 be such a set. By the complexity analysis for Σ-query entailment for *DL-Lite*$^{\mathcal{N}}_{bool}$ in Section 6.2, for each $\boldsymbol{t} \in \Xi_0$ there exists $\Xi_{\boldsymbol{t}} \subseteq \Xi_0$ such that $\{\boldsymbol{t}\}\cup\Xi_{\boldsymbol{t}}$ is $\mathcal{T}$-precisely realisable. Take the disjoint union of models of $\mathcal{T}$ realising $\{\boldsymbol{t}\}\cup\Xi_{\boldsymbol{t}}$, for $\boldsymbol{t} \in \Xi_0$. It is readily seen that this is a model of $\mathcal{T}$ precisely realising Ξ_0. ❑

Acknowledgements. We thank Marco Benedetti, Luca Pulina, Uli Sattler, Thomas Schneider and Petra Selmer for their help in conducting experiments described in Section 9.

References

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003.

[2] N. Noy, M. Musen, Promptdiff: a fixed-point algorithm for comparing ontology versions, in: Proc. of the 18th Nat. Conf. on Artificial Intelligence (AAAI 2002), 2002, pp. 744–750.

[3] B. Cuenca Grau, I. Horrocks, Y. Kazakov, U. Sattler, A logical framework for modularity of ontologies, in: Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007), 2007, pp. 298–303.

[4] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, *DL-Lite*: Tractable description logics for ontologies, in: Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005), 2005, pp. 602–607.

[5] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Data complexity of query answering in description logics, in: Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006), 2006, pp. 260–270.

[6] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family, J. of Automated Reasoning 39 (3) (2007) 385–429.

[7] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, Linking data to ontologies, J. on Data Semantics X (2008) 133–173.

[8] A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, M. Zakharyaschev, Reasoning over Extended ER models, in: Proc. of the 26th Int. Conf. on Conceptual Modeling (ER 2007), Vol. 4801 of Lecture Notes in Computer Science, Springer, 2007, pp. 277–292.

[9] A. Artale, D. Calvanese, R. Kontchakov, M. Zakharyaschev, *DL-Lite* in the light of first-order logic, in: Proc. of the 22nd Nat. Conf. on Artificial Intelligence (AAAI 2007), 2007, pp. 361–366.

[10] A. Artale, D. Calvanese, R. Kontchakov, M. Zakharyaschev, The *DL-Lite* family and their relations, Tech. Rep. BBKCS-09-03, School of Computer Sci and Inf Systems, Birkbeck College, to appear in JAIR (April 2009).

[11] H. Stuckenschmidt, C. Parent, S. Spaccapietra (Eds.), Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization, Vol. 5445 of Lecture Notes in Computer Science, Springer, 2009.

[12] P. Haase, V. Honavar, O. Kutz, Y. Sure, A. Tamilin (Eds.), Proceedings of the 1st International Workshop on Modular Ontologies, WoMO'06, co-located with the International Semantic Web Conference, ISWC'06 November 5, 2006, Athens, Georgia, USA, Vol. 232 of CEUR Workshop Proceedings, CEUR-WS.org, 2007.

[13] B. Cuenca Grau, V. Honavar, A. Schlicht, F. Wolter (Eds.), Proceedings of the 2nd International Workshop on Modular Ontologies, WoMO 2007, Whistler, Canada, October 28, 2007, Vol. 315 of CEUR Workshop Proceedings, CEUR-WS.org, 2008.

[14] N. F. Noy, M. A. Musen, Specifying ontology views by traversal, in: S. A. McIlraith, D. Plexousakis, F. van Harmelen (Eds.), Proc. of the 3rd Int. Semantic Web Conf. (ISWC 2004), Vol. 3298 of Lecture Notes in Computer Science, Springer, 2004, pp. 713–725.

[15] J. Seidenberg, A. Rector, Web ontology segmentation: analysis, classification and use, in: L. Carr, D. D. Roure, A. Iyengar, C. A. Goble, M. Dahlin (Eds.), Proc. of the 15th Int. Conf. on World Wide Web, WWW 2006, ACM Press, New York, NY, USA, 2006, pp. 13–22.

[16] B. Cuenca Grau, B. Parsia, E. Sirin, A. Kalyanpur, Modularity and web ontologies, in: P. Doherty, J. Mylopoulos, C. A. Welty (Eds.), Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006), AAAI Press, 2006, pp. 198–209.

[17] B. Cuenca Grau, I. Horrocks, Y. Kazakov, U. Sattler, Modular reuse of ontologies: Theory and practice, J. of Artificial Intelligence Research 31 (2008) 273–318.

[18] B. Konev, C. Lutz, D. Walther, F. Wolter, Semantic modularity and module extraction in description logics, in: M. Ghallab, C. D. Spyropoulos, N. Fakotakis, N. Avouris (Eds.), Proceedings of the 18th European Conference on Artificial Intelligence (ECAI08), Vol. 178 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2008, pp. 55–59.

[19] J. Lang, P. Liberatore, P. Marquis, Propositional independence: Formula-variable independence and forgetting, J. of Artificial Intelligence Research 18 (2003) 391–443.

[20] F. Lin, R. Reiter, Forget it!, in: Proceedings of the AAAI Fall Symposium on Relevance, 1994, pp. 154–159.

[21] T. Eiter, K. Wang, Semantic forgetting in answer set programming, Artificial Intelligence 172 (3) (2008) 1644–1672.
[22] A. Pitts, On an interpretation of second-order quantification in first-order intuitionistic propositional logic, J. Symbolic Logic 57 (1) (1992) 33–52.
[23] A. Visser, Uniform interpolation and layered bisimulation, in: P. Hájek (Ed.), Gödel'96, Vol. 6 of Lecture Notes in Logic, Springer, 1996, pp. 139–164.
[24] S. Ghilardi, C. Lutz, F. Wolter, Did I damage my ontology? A case for conservative extensions in description logic, in: Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006), 2006, pp. 187–197.
[25] B. Konev, C. Lutz, D. Walther, F. Wolter, Formal properties of modularisation, in: Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization, Springer, 2009, pp. 25–66.
[26] B. Konev, D. Walther, F. Wolter, Forgetting and uniform interpolation in large-scale description logic terminologies, in: Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009), 2009, pp. 830–835.
[27] Z. Wang, K. Wang, R. W. Topor, J. Z. Pan, Forgetting concepts in *DL-Lite*, in: S. Bechhofer, M. Hauswirth, J. Hoffmann, M. Koubarakis (Eds.), Proc. of the 5th Eur. Semantic Web Conf. (ESWC 2008), Vol. 5021 of Lecture Notes in Computer Science, Springer, 2008, pp. 245–257.
[28] L. Pulina, A. Tacchella, A self-adaptive multi-engine solver for quantified Boolean formulas, Constraints 14 (1) (2009) 80–116.
[29] R. Kontchakov, F. Wolter, M. Zakharyaschev, Can you tell the difference between *DL-Lite* ontologies?, in: Proc. of the 11th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2008), 2008, pp. 285–295.
[30] R. Kontchakov, L. Pulina, U. Sattler, T. Schneider, P. Seimer, F. Wolter, M. Zakharyaschev, Minimal module extraction from *DL-Lite* ontologies using QBF solvers, in: Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009), 2009, pp. 836–840.
[31] A. Acciarri, D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, M. Palmieri, R. Rosati, QuOnto: Querying ontologies, in: Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005), 2005, pp. 1670–1671.
[32] A. Poggi, M. Rodriguez, M. Ruzzi, Ontology-based database access with DIG-Mastro and the OBDA Plugin for Protégé, in: K. Clark, P. F. Patel-Schneider (Eds.), Proc. of the 4th Int. Workshop on OWL: Experiences and Directions (OWLED 2008 DC), 2008.
[33] C. Lutz, D. Walther, F. Wolter, Conservative extensions in expressive description logics, in: Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007), 2007, pp. 453–458.
[34] C. Lutz, F. Wolter, Mathematical logic for life science ontologies, in: H. Ono, M. Kanazawa, R. J. G. B. de Queiroz (Eds.), Logic, Language, Information and Computation, 16th Int. Workshop, WoLLIC 2009, Vol. 5514 of Lecture Notes in Computer Science, Springer, 2009, pp. 37–47.
[35] R. Diaconescu, J. Goguen, P. Stefaneas, Logical support for modularisation, in: G. Huet, G. Plotkin (Eds.), Logical Environments, Cambridge University Press, New York, 1993, pp. 83–130.
[36] P. Mosses (Ed.), CASL Reference Manual: The Complete Documentation Of The Common Algebraic Specification Language, Vol. 2960 of Lecture Notes in Computer Science, Springer, 2004.
[37] P. Byers, D. H. Pitt, Conservative extensions: a cautionary note, Bulletin of the EATCS 41 (1990) 196–201.
[38] T. Maibaum, Conservative extensions, interpretations between theories and all that!, in: M. Bidoit, M. Dauchet (Eds.), Proc. of the 7th Int. Joint Conf. CAAP/FASE on Theory and Practice of Software Development, TAPSOFT '97, Vol. 1214 of Lecture Notes in Computer Science, Springer, 1997, pp. 40–66.
[39] G. Antoniou, A. Kehagias, A note on the refinement of ontologies, Int. J. of Intelligent Systems 15 (7) (2000) 623–632.
[40] T. Eiter, M. Fink, S. Woltran, Semantical characterizations and complexity of equivalences in answer set programming, ACM Trans. Comput. Log. 8 (3).
[41] M. Fink, Equivalences in answer-set programming by countermodels in the logic of here-and-there, in: M. G. de la Banda, E. Pontelli (Eds.), Proc. of the 24th Int. Conf. on Logic Programming (ICLP 2008), Vol. 5366 of Lecture Notes in Computer Science, Springer, 2008, pp. 99–113.
[42] D. Pearce, A. Valverde, Synonymous theories in answer set programming and equilibrium logic, in: Proc. of the 16th European Conf. on Artificial Intelligence (ECAI 2004), 2004, pp. 388–392.
[43] V. Lifschitz, D. Pearce, A. Valverde, A characterization of strong equivalence for logic programs with variables, in: Proc. of the 9th Int. Conf. on Logic Programming and Nonmonotonic Reasoning (LPNMR), 2007, pp. 188–200.
[44] C. Lutz, F. Wolter, Deciding inseparability and conservative extensions in the description logic EL, Journal of Symbolic Computation(to appear).
[45] C. Chang, H. Keisler, Model Theory, Elsevier, 1990.
[46] C. Areces, B. ten Cate, Hybrid logics, in: P. Blackburn, J. van Benthem, F. Wolter (Eds.), Handbook of Modal Logic, Elsevier, 2006, pp. 821–868.
[47] C. Papadimitriou, Computational Complexity, Addison-Wesley, 1994.
[48] D. Kozen, Theory of Computation, Springer, 2006.
[49] H. Kleine Büning, T. Lettman, Propositional logic: Deduction and algorithms, Cambridge University Press, 1999.
[50] C. Lutz, F. Wolter, Conservative extensions in the lightweight description logic EL, in: F. Pfenning (Ed.), Proc. of the 21st Conf. on Automated Deduction (CADE-21), Vol. 4603 of Lecture Notes in Computer Science, Springer, 2007, pp. 84–99.
[51] S. Feferman, R. L. Vaught, The first-order properties of algebraic systems, Fundamenta Mathematicae 47 (1959) 57–103.
[52] V. Kuncak, H. H. Nguyen, M. C. Rinard, Deciding Boolean algebra with Presburger arithmetic, J. of Automated Reasoning 36 (3) (2006) 213–239.
[53] S. Ghilardi, C. Lutz, F. Wolter, M. Zakharyaschev, Conservative extensions in modal logics, in: G. Governatori, I. Hodkinson, Y. Venema (Eds.), Advances in Modal Logics Volume 6, College Publications, 2006, pp. 187–207.
[54] H. Liu, C. Lutz, M. Milicic, F. Wolter, Updating description logic aboxes, in: Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006), 2006, pp. 46–56.
[55] A. Remshagen, K. Truemper, The complexity of futile questioning, in: H. R. Arabnia, P. L. Zhou (Eds.), Proc. of the Int. Conf. on Foundations of Computer Science (FCS 2007), CSREA Press, 2007, pp. 132–138.

[56] M. Benedetti, sKizzo: A suite to evaluate and certify QBFs, in: R. Nieuwenhuis (Ed.), Proc. the 20th Int. Conf. on Automated Deduction (CADE–20), Vol. 3632 of Lecture Notes in Computer Science, Springer, 2005, pp. 369–376.

[57] H. Samulowitz, F. Bacchus, Binary clause reasoning in QBF, in: A. Biere, C. P. Gomes (Eds.), Proc. the 9th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT 2006), Vol. 4121 of Lecture Notes in Computer Science, Springer, 2006, pp. 353–367.

[58] L. Zhang, S. Malik, Towards a symmetric treatment of satisfaction and conflicts in quantified boolean formula evaluation, in: P. V. Hentenryck (Ed.), Proc. of the 8th Int. Conf. on Principles and Practice of Constraint Programming (CP 2002), Vol. 2470 of Lecture Notes in Computer Science, Springer, 2002, pp. 200–215.

[59] L. Zhang, S. Malik, Conflict driven learning in a quantified boolean satisfiability solver, in: L. T. Pileggi, A. Kuehlmann (Eds.), Proc. of the IEEE/ACM Int. Conf. on Computer-aided Design (ICCAD), ACM, 2002, pp. 442–449.

[60] E. Giunchiglia, M. Narizzano, A. Tacchella, Clause-term resolution and learning in quantified Boolean logic satisfiability, J. of Artificial Intelligence Research 26 (2006) 371–416.

[61] R. Rosati, Finite model reasoning in *DL-Lite*, in: S. Bechhofer, M. Hauswirth, J. Hoffmann, M. Koubarakis (Eds.), Proc. of the 5th Eur. Semantic Web Conf. (ESWC 2008), Vol. 5021 of Lecture Notes in Computer Science, Springer, 2008, pp. 215–229.