View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Birkbeck Institutional Research Online

Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

When Are Description Logic Knowledge Bases Indistinguishable?*

E. Botoeva,[1] **R. Kontchakov,**[3] **V. Ryzhikov,**[1] **F. Wolter**[2] **and M. Zakharyaschev**[3]

[1]Faculty of Computer Science
Free University of Bozen-Bolzano, Italy
{botoeva,ryzhikov}@inf.unibz.it

[2]Dept. of Computer Science
University of Liverpool, UK
wolter@liverpool.ac.uk

[3]Dept. of Computer Science
Birkbeck, London, UK
{roman,michael}@dcs.bbk.ac.uk

Abstract

Deciding inseparability of description logic knowledge bases (KBs) with respect to conjunctive queries is fundamental for many KB engineering and maintenance tasks including versioning, module extraction, knowledge exchange and forgetting. We study the combined and data complexity of this inseparability problem for fragments of *Horn*-$\mathcal{ALCHI}$, including the description logics underpinning *OWL 2 QL* and *OWL 2 EL*.

1 Introduction

A description logic (DL) knowledge base (KB) consists of a terminological box (TBox) and an assertion box (ABox). The TBox represents conceptual knowledge by providing a vocabulary for a domain of interest together with axioms that describe semantic relationships between the vocabulary items. To illustrate, the following toy TBox, $\mathcal{T}_a$, defines a vocabulary for the automotive industry:

$$\begin{array}{l} \mathit{Minivan} \sqsubseteq \mathit{Automobile}, \qquad \mathit{Hybrid} \sqsubseteq \mathit{Automobile},\\ \mathit{Automobile} \sqsubseteq \exists \mathit{poweredBy}.\mathit{Engine},\\ \mathit{Hybrid} \sqsubseteq \exists \mathit{poweredBy}.\mathit{EEngine} \sqcap \exists \mathit{poweredBy}.\mathit{ICEngine},\\ \mathit{EEngine} \sqsubseteq \mathit{Engine}, \qquad \mathit{ICEngine} \sqsubseteq \mathit{Engine}. \end{array}$$

For example, the first two axioms say that minivans and hybrids are automobiles; the third axiom claims that every automobile is powered by an engine. Thus, the TBox introduces, among others, *concept names* (sets) *Minivan*, *Automobile* and *Engine*, states that the concept *Minivan* is subsumed by the concept *Automobile* and uses the *role name* (binary relation) *poweredBy* to say that automobiles are powered by engines. The last two axioms state that electric and internal combustion engines are engines. TBoxes are often called *ontologies* and presented in applications in terms of the Web Ontology Language OWL 2, which is underpinned by DLs.

The ABox of a KB is a set of facts storing data about the concept and role names introduced in the TBox. An example ABox, $\mathcal{A}_a$, in the automotive domain is given by

$$\begin{array}{l} \mathit{Hybrid}(\mathsf{toyota\_highlander}),\\ \quad \mathit{Minivan}(\mathsf{toyota\_highlander}), \ \ \mathit{Minivan}(\mathsf{nissan\_note}). \end{array}$$

Typical applications of KBs in modern information systems use the semantics of concepts and roles in the TBox to enable the user to query the data in the ABox. This is particularly useful if the data is incomplete or comes from heterogenous data sources which is the case, for example, in linked data applications [Polleres *et al.*, 2013] and large scale data integration projects [Poggi *et al.*, 2008; Giese *et al.*, 2013], or if the data comprises web content gathered by search engines using semantic markup [Hitzler *et al.*, 2009].

As the data may be incomplete, the open world assumption is made when querying a KB $\mathcal{K}$: a tuple $\boldsymbol{a}$ of individuals from $\mathcal{K}$ is a (*certain*) *answer* to a query $\boldsymbol{q}$ over $\mathcal{K}$ iff $\boldsymbol{q}(\boldsymbol{a})$ is true in every model $\mathcal{I}$ of $\mathcal{K}$. As general first-order queries are undecidable under the open-world semantics, the basic and most important querying instrument is conjunctive queries (CQs), which are ubiquitous in relational database systems and form the core of the Semantic Web query language SPARQL. A CQ $\boldsymbol{q}(\boldsymbol{x})$ is a first-order formula $\exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$, where $\varphi(\boldsymbol{x}, \boldsymbol{y})$ is a conjunction of atoms of the form $A(z_1)$ or $P(z_1, z_2)$ for a concept name A, a role name P, and variables z_1, z_2 from $\boldsymbol{x}, \boldsymbol{y}$.[1] For instance, to find minivans powered by electric engines, one can use the following CQ:

$$\boldsymbol{q}(x) = \exists y\, (\mathit{Minivan}(x) \land \mathit{poweredBy}(x, y) \land \mathit{EEngine}(y)).$$

Then $\mathsf{toyota\_highlander}$ is its only certain answer in $(\mathcal{T}_a, \mathcal{A}_a)$.

The problem of answering CQs over KBs has been the focus of significant research in the DL community with deep complexity results for a large variety of DLs (see below), the introduction of new DLs for which query answering is tractable for data complexity [Hustadt *et al.*, 2005; Calvanese *et al.*, 2007], the invention of various query answering techniques [Calvanese *et al.*, 2007; Lutz *et al.*, 2009] and the development of powerful implemented systems; see, e.g., [Kontchakov and Zakharyaschev, 2014] and references therein.

Apart from developing query answering techniques, a major research problem is KB engineering and maintenance. In

*This paper was invited for submission to the Best Papers From Sister Conferences Track, based on a paper that appeared in the Proc. of the 14th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2014).

[1]Since we consider Horn DLs, the results of this paper actually apply to disjunctions (or unions) of CQs (UCQs). For simplicity, however, we concentrate on CQs only.

fact, with typically large data and often complex and tangled ontologies, tool support for transforming and comparing KBs is becoming indispensable for applications. To begin with, KBs are never static entities. Like most software artefacts, they are updated to incorporate new information, and distinct versions are introduced for different applications. Thus, developing support for KB *versioning* has become a major research problem [Jiménez-Ruiz *et al.*, 2011; Konev *et al.*, 2012]. As dealing with a large and semantically tangled KB can be costly, one may want to *extract* from it a smaller *module* that is indistinguishable from the whole KB as far as the given application is concerned [Stuckenschmidt *et al.*, 2009]. Another technique for extracting relevant information is *forgetting*, where the task is to replace a given KB by a new KB using only those concept and role names that are needed by the application but still providing the same information about those names as the original KB [Konev *et al.*, 2009; Koopmann and Schmidt, 2014b]. Finally, the vocabulary of a given KB may not be convenient for a new application. In this case, similarly to *data exchange* in databases [Arenas *et al.*, 2014]—where data structured under a source schema is converted to data under a target schema—one may want to transform a KB in a source signature to a KB given in a more useful target signature and representing the original KB in an accurate way. This task is known as *knowledge exchange* [Arenas *et al.*, 2012; 2013].

In this paper, we investigate a relationship between KBs that is fundamental for all such tasks if querying the data via CQs is the main application. Let Σ be a signature consisting of a set of concept and role names. We say that KBs $\mathcal{K}_1$ and $\mathcal{K}_2$ are Σ-*query inseparable* and write $\mathcal{K}_1 \equiv_\Sigma \mathcal{K}_2$ if any CQ formulated in Σ has the same answers over $\mathcal{K}_1$ and $\mathcal{K}_2$. Note that even for Σ containing all concept and role names in the KBs, Σ-query inseparability does not necessarily imply logical equivalence: e.g., $(\emptyset, \{A(a)\})$ is $\{A, B\}$-query inseparable from $(\{B \sqsubseteq A\}, \{A(a)\})$ but the two KBs are clearly not logically equivalent. Thus, if KBs are used for purposes other than querying data via CQs, then different notions of inseparability are required. We now discuss the applications of Σ-query inseparability for the tasks above in more detail.

Versioning. Version control systems for KBs provide a range of operations including, for example, computing the relevant differences between KBs, merging KBs and recovering KBs. All these operations rely on checking whether two versions, $\mathcal{K}_1$ and $\mathcal{K}_2$, of a KB are indistinguishable from the application point of view. If that application is querying the data using CQs over a given signature Σ, then $\mathcal{K}_1$ and $\mathcal{K}_2$ should be regarded as indistinguishable just in case they give the same answers to CQs in Σ. Thus, the basic task for a query-centric approach to KB versioning is to check whether $\mathcal{K}_1 \equiv_\Sigma \mathcal{K}_2$.

Modularisation. Modularisation and module extraction are major research topics in ontology engineering and maintenance. In module extraction, the problem is to find a (small) subset of the axioms of a given large KB that is indistinguishable from it with respect to the intended application. If that application is querying a KB $\mathcal{K}$ using CQs over a signature Σ, then the problem is to find a small Σ-*query module of* $\mathcal{K}$, that is, a KB $\mathcal{K}' \subseteq \mathcal{K}$ with $\mathcal{K}' \equiv_\Sigma \mathcal{K}$. Note that one can extract a minimal Σ-query module from a KB using a polynomial-time algorithm with the Σ-query inseparability check as an oracle. To illustrate the notion of Σ-query module, consider the automotive ontology $\mathcal{K}_a = (\mathcal{T}_a, \mathcal{A}_a)$ described above and the signature $\Sigma_m = \{\mathit{Automobile}, \mathit{Engine}, \mathit{poweredBy}\}$. Then $\mathcal{K}_m = (\mathcal{T}_m, \mathcal{A}_a)$ is a Σ_m-query module of $\mathcal{K}_a$, where $\mathcal{T}_m$ is

$$\mathit{Minivan} \sqsubseteq \mathit{Automobile}, \ \mathit{Automobile} \sqsubseteq \exists \mathit{poweredBy}.\mathit{Engine}.$$

Knowledge Exchange. In knowledge exchange, we want to transform a KB $\mathcal{K}_1$ in a signature Σ_1 to a KB $\mathcal{K}_2$ in a new signature Σ_2 connected to Σ_1 via a declarative mapping specification given by a TBox $\mathcal{T}_{12}$. Such mapping specifications between KBs are also known as ontology alignments or ontology matchings and have been studied extensively [Shvaiko and Euzenat, 2013]. If, as above, we are interested in querying data via CQs, then the target KB $\mathcal{K}_2$ should be a sound and complete representation of $\mathcal{K}_1$ w.r.t. querying data, and so satisfy the condition $\mathcal{K}_1 \cup \mathcal{T}_{12} \equiv_{\Sigma_2} \mathcal{K}_2$, in which case it is called a *universal CQ-solution*. To illustrate, consider again the ontology $\mathcal{K}_a = (\mathcal{T}_a, \mathcal{A}_a)$, and let $\mathcal{T}_{ae}$ relate the signature Σ_a of $\mathcal{K}_a$ to $\Sigma_e = \{\mathit{Car}, \mathit{HybridCar}, \mathit{ElectricMotor}, \mathit{Motor}, \mathit{hasMotor}\}$:

$$\begin{array}{rr} \mathit{Automobile} \sqsubseteq \mathit{Car}, & \mathit{Hybrid} \sqsubseteq \mathit{HybridCar}, \\ \mathit{Engine} \sqsubseteq \mathit{Motor}, & \mathit{EEngine} \sqsubseteq \mathit{ElectricMotor}, \\ & \mathit{poweredBy} \sqsubseteq \mathit{hasMotor}. \end{array}$$

Then $\mathcal{K}_e = (\mathcal{T}_e, \mathcal{A}_e)$ is a universal CQ-solution, where

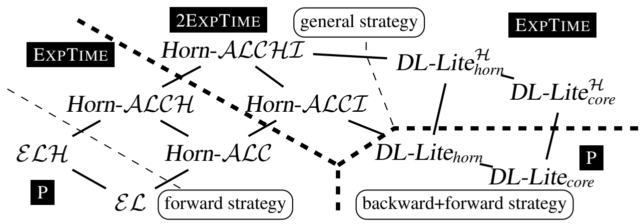
$$\begin{aligned} \mathcal{T}_e = \{ & \mathit{ElectricMotor} \sqsubseteq \mathit{Motor}, \ \mathit{Car} \sqsubseteq \exists \mathit{hasMotor}.\mathit{Motor}, \\ & \mathit{HybridCar} \sqsubseteq \mathit{Car} \sqcap \exists \mathit{hasMotor}.\mathit{ElectricMotor} \}, \\ \mathcal{A}_e = \{ & \mathit{HybridCar}(\mathsf{toyota\_highlander}), \mathit{Car}(\mathsf{nissan\_note}) \}. \end{aligned}$$

Forgetting. A KB $\mathcal{K}'$ results from *forgetting* a signature Σ in a KB $\mathcal{K}$ if $\mathcal{K}' \equiv_{\mathsf{sig}(\mathcal{K}) \setminus \Sigma} \mathcal{K}$ and $\mathsf{sig}(\mathcal{K}') \subseteq \mathsf{sig}(\mathcal{K}) \setminus \Sigma$, where $\mathsf{sig}(\mathcal{K})$ is the signature of $\mathcal{K}$. Thus, the result of forgetting Σ does not use Σ and gives the same answers to CQs without symbols in Σ as $\mathcal{K}$. The result of forgetting is also called a *uniform interpolant* for $\mathcal{K}$ w.r.t. $\mathsf{sig}(\mathcal{K}) \setminus \Sigma$. Forgetting is of interest for a number of applications. Typically, when reusing an existing KB in a new application, only a small number of its symbols is relevant, and so instead of reusing the whole KB, one can take the potentially smaller KB resulting from forgetting the extraneous symbols. Forgetting can also be used for *predicate hiding*: if a KB is to be published, but some part of it has to be concealed from the public, then this part can be removed by forgetting its symbols [Cuenca Grau and Motik, 2012]. Finally, forgetting can be used for *KB summary*: the result of forgetting often provides a smaller and more focused ontology that summarises what the original ontology says about the retained symbols, potentially facilitating KB comprehension. To illustrate, for $\Sigma_f = \{\mathit{Automobile}, \mathit{Engine}, \mathit{poweredBy}\}$, the KB $(\mathcal{T}_f, \mathcal{A}_f)$,

$$\begin{aligned} \mathcal{T}_f &= \{ \mathit{Automobile} \sqsubseteq \exists \mathit{poweredBy}.\mathit{Engine} \}, \\ \mathcal{A}_f &= \{ \mathit{Automobile}(\mathsf{toyota\_highlander}), \\ & \qquad \mathit{Automobile}(\mathsf{nissan\_note}) \}, \end{aligned}$$

is a result of forgetting $\mathsf{sig}(\mathcal{K}_a) \setminus \Sigma_f$ in $\mathcal{K}_a$.

We investigate the data and combined complexity of deciding Σ-query inseparability of KBs given in various fragments of the DL *Horn*-$\mathcal{ALCHI}$ [Krötzsch *et al.*, 2013], which include *DL-Lite*$^{\mathcal{H}}_{core}$ [Calvanese *et al.*, 2007; Artale *et al.*, 2009], $\mathcal{EL}$ and $\mathcal{ELH}$ [Baader *et al.*, 2005] underlying the OWL 2 profiles *OWL 2 QL* and *OWL 2 EL*. For all of these DLs, Σ-query inseparability turns out to be P-complete for data complexity, which matches the data complexity of CQ evaluation in all of our DLs lying outside the *DL-Lite* family. The obtained tight combined complexity results are summarised in the diagram below:



Most interesting are ExpTime- and 2ExpTime-completeness of *DL-Lite*$^{\mathcal{H}}_{core}$ and *Horn*-$\mathcal{ALCI}$, respectively, which contrast with NP- and ExpTime-completeness of CQ evaluation in these logics. For *DL-Lite* without role inclusions and $\mathcal{ELH}$, Σ-query inseparability is P-complete, while CQ evaluation is NP-complete. In general, it is the combined presence of inverse roles and qualified existential restrictions (or role inclusions) that makes Σ-query inseparability hard. To establish the upper complexity bounds, we develop a uniform game-theoretic framework for checking finite Σ-homomorphic embeddability between (possibly infinite) materialisations of KBs. All omitted proofs can be found in the full version at http://tinyurl.com/poa49vf.

2 *Horn*-$\mathcal{ALCHI}$ and its Fragments

All the DLs considered in this paper are Horn fragments of $\mathcal{ALCHI}$. To define them, we fix lists of *individual names* a_i, *concept names* A_i, and *role names* P_i, for $i < \omega$. A *role* is a role name P_i or an *inverse role* P_i^-; we assume $(P_i^-)^- = P_i$. $\mathcal{ALCI}$-*concepts*, C, are defined by the grammar

$$C \ ::= \ A_i \ \mid \ \top \ \mid \ \neg C \ \mid \ C_1 \sqcap C_2 \ \mid \ \exists R.C,$$

where R is a role. We use $\bot$, $C_1 \sqcup C_2$ and $\forall R.C$ as abbreviations for $\neg\top$, $\neg(\neg C_1 \sqcap \neg C_2)$ and $\neg\exists R.\neg C$, respectively. $\mathcal{ALC}$-*concepts* are $\mathcal{ALCI}$-concepts without inverse roles; $\mathcal{EL}$-*concepts* are $\mathcal{ALC}$-concepts without $\neg$. *DL-Lite*$_{horn}$-*concepts* are $\mathcal{ALCI}$-concepts without $\neg$, in which $C = \top$ in every occurrence of $\exists R.C$. Finally, *DL-Lite*$_{core}$-*concepts* are *DL-Lite*$_{horn}$-concepts without $\sqcap$; in other words, they are *basic concepts* of the form $\top$, A_i or $\exists R$ (a shortcut for $\exists R.\top$).

For a DL $\mathcal{L}$, an $\mathcal{L}$-*concept inclusion* (CI) takes the form $C \sqsubseteq D$, where C and D are $\mathcal{L}$-concepts. An $\mathcal{L}$-*TBox*, $\mathcal{T}$, contains a finite set of $\mathcal{L}$-CIs. An $\mathcal{ALCHI}$, *DL-Lite*$^{\mathcal{H}}_{horn}$ and *DL-Lite*$^{\mathcal{H}}_{core}$ TBox can also contain a finite set of *role inclusions* (RIs) $R_1 \sqsubseteq R_2$, where the R_i are roles. In $\mathcal{ALCH}$ and $\mathcal{ELH}$, TBoxes have RIs but without inverse roles. *DL-Lite* TBoxes also contain *disjointness constraints* $B_1 \sqcap B_2 \sqsubseteq \bot$ and $R_1 \sqcap R_2 \sqsubseteq \bot$, for basic concepts B_i and roles R_i.[2]

To introduce the Horn fragments of these DLs, we require the following (standard) recursive definition [Hustadt *et al.*, 2005; Kazakov, 2009]: a concept C occurs positively in C; if C occurs positively (respectively, negatively) in C' then C occurs positively (negatively) in $C' \sqcap D$, $\exists R.C'$, $D \sqsubseteq C'$, and it occurs negatively (positively) in $\neg C'$ and $C' \sqsubseteq D$. A TBox $\mathcal{T}$ is *Horn* if no concept of the form $\neg C$ occurs negatively in $\mathcal{T}$ and no $\exists R.\neg C$ occurs positively in $\mathcal{T}$. In the DL *Horn*-$\mathcal{L}$, where $\mathcal{L}$ is one of our DLs, only *Horn*-$\mathcal{L}$-TBoxes are allowed. Clearly, $\mathcal{EL}$- and *DL-Lite*-TBoxes are Horn by definition.

An *ABox*, $\mathcal{A}$, is a finite set of *assertions* of the form $A_k(a_i)$ or $P_k(a_i, a_j)$. An $\mathcal{L}$-TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ form an $\mathcal{L}$-KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$; $\mathsf{ind}(\mathcal{K})$ is the set of individual names in $\mathcal{K}$.

The semantics for the DLs is defined in the usual way based on interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ that comply with the *unique name assumption*: $a_i^{\mathcal{I}} \neq a_j^{\mathcal{I}}$ for $i \neq j$ [Baader *et al.*, 2003]. We write $\mathcal{I} \models \alpha$ in case an inclusion or assertion α is true in $\mathcal{I}$. If $\mathcal{I} \models \alpha$, for all $\alpha \in \mathcal{T} \cup \mathcal{A}$, then $\mathcal{I}$ is a *model* of a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$; in symbols: $\mathcal{I} \models \mathcal{K}$. $\mathcal{K}$ is *consistent* if it has a model. $\mathcal{K} \models \alpha$ means that $\mathcal{I} \models \alpha$ for all $\mathcal{I} \models \mathcal{K}$.

A *conjunctive query* (CQ) $\boldsymbol{q}(\boldsymbol{x})$ is a formula $\exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$, where φ is a conjunction of atoms of the form $A_k(z_1)$ or $P_k(z_1, z_2)$ with z_i in $\boldsymbol{x}, \boldsymbol{y}$. A tuple $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{K})$ (of the same length as $\boldsymbol{x}$) is a *certain answer* to $\boldsymbol{q}(\boldsymbol{x})$ over $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I} \models \boldsymbol{q}(\boldsymbol{a})$ for all $\mathcal{I} \models \mathcal{K}$; in this case we write $\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$. If $\boldsymbol{x} = \emptyset$, the answer to $\boldsymbol{q}$ is 'yes' if $\mathcal{K} \models \boldsymbol{q}$ and 'no' otherwise.

For combined complexity, the problem '$\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$?' is NP-complete for the *DL-Lite* logics [Calvanese *et al.*, 2007], $\mathcal{EL}$ and $\mathcal{ELH}$ [Rosati, 2007], and ExpTime-complete for the remaining Horn DLs above [Eiter *et al.*, 2008]. For data complexity (with fixed $\mathcal{T}$ and $\boldsymbol{q}$), this problem is in AC^0 for the *DL-Lite* logics [Calvanese *et al.*, 2007] and P-complete for the remaining DLs [Rosati, 2007; Eiter *et al.*, 2008].

A *signature*, Σ, is a set of concept and role names. By a Σ-concept, Σ-role, Σ-CQ, etc. we understand any concept, role, CQ, etc. constructed using the names from Σ. Given an interpretation $\mathcal{I}$ and a signature Σ, we define the Σ-*types* $\boldsymbol{t}^{\mathcal{I}}_{\Sigma}(u)$ and $\boldsymbol{r}^{\mathcal{I}}_{\Sigma}(u, v)$ of $u, v \in \Delta^{\mathcal{I}}$ by taking:

$$\boldsymbol{t}^{\mathcal{I}}_{\Sigma}(u) = \{\, \Sigma\text{-concept name } A \mid u \in A^{\mathcal{I}} \,\},$$
$$\boldsymbol{r}^{\mathcal{I}}_{\Sigma}(u, v) = \{\, \Sigma\text{-role } R \mid (u, v) \in R^{\mathcal{I}} \,\}.$$

3 Σ-Query Entailment and Inseparability

Now we define the central notions of the paper, Σ-query entailment and inseparability, and establish their semantic characterisation based on the notion of materialisation. We then show how to construct materialisations by developing a theory of finitely generated materialisations.

Definition 1. Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be KBs and Σ a signature. We say that $\mathcal{K}_1$ Σ-*query entails* $\mathcal{K}_2$ if $\mathcal{K}_2 \models \boldsymbol{q}(\boldsymbol{a})$ implies $\mathcal{K}_1 \models \boldsymbol{q}(\boldsymbol{a})$, for *all* Σ-CQs $\boldsymbol{q}(\boldsymbol{x})$ and all tuples $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{K}_2)$. Knowledge bases $\mathcal{K}_1$ and $\mathcal{K}_2$ are Σ-*query inseparable* if they Σ-query entail each other; in this case we write $\mathcal{K}_1 \equiv_{\Sigma} \mathcal{K}_2$.

Checking Σ-query inseparability can be trivially reduced to two Σ-query entailment checks. Conversely, for most languages we have a semantically transparent reduction of Σ-query entailment to Σ-query inseparability:

Theorem 1. *Let $\mathcal{L}$ be any of our DLs containing $\mathcal{EL}$ or having role inclusions. Then Σ-query entailment of $\mathcal{L}$-KBs is* LogSpace*-reducible to Σ-query inseparability of $\mathcal{L}$-KBs.*

[2] Although role disjointness constraints are not in the syntax of $\mathcal{ALCHI}$, they play no essential part in our constructions, and the techniques we develop for $\mathcal{ALCHI}$ are also applicable to *DL-Lite*.

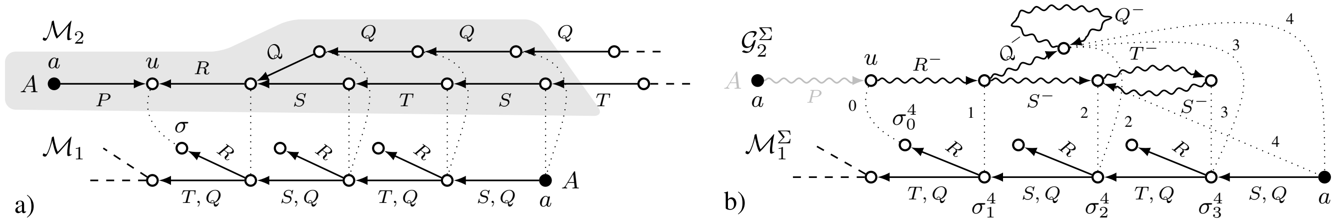


Figure 1: a) materialisations $\mathcal{M}_1$ and $\mathcal{M}_2$ of $\mathcal{K}_1$ and $\mathcal{K}_2$ and b) a 4-winning (backward) strategy in $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$.

To prove the upper complexity bounds for both problems, we give a semantic characterisation of Σ-query entailment in terms of materialisations and finite Σ-homomorphisms. An interpretation $\mathcal{M}$ is a *materialisation* of a KB $\mathcal{K}$ if $\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{M} \models \boldsymbol{q}(\boldsymbol{a})$, for all CQs $\boldsymbol{q}(\boldsymbol{x})$ and all tuples $\boldsymbol{a}$ in $\mathsf{ind}(\mathcal{K})$. We say that $\mathcal{K}$ is *materialisable* if it has a materialisation.

Suppose $\mathcal{M}_i$ is a materialisation of $\mathcal{K}_i$, for $i = 1, 2$. A function $h\colon \Delta^{\mathcal{M}_2} \to \Delta^{\mathcal{M}_1}$ is a Σ-*homomorphism* if $\boldsymbol{t}_\Sigma^{\mathcal{M}_2}(u) \subseteq \boldsymbol{t}_\Sigma^{\mathcal{M}_1}(h(u))$ and $\boldsymbol{r}_\Sigma^{\mathcal{M}_2}(u, v) \subseteq \boldsymbol{r}_\Sigma^{\mathcal{M}_1}(h(u), h(v))$, for all $u, v \in \Delta^{\mathcal{M}_2}$, and $h(a^{\mathcal{M}_2}) = a^{\mathcal{M}_1}$ for all $a \in \mathsf{ind}(\mathcal{K}_2)$ that belong to a Σ-concept or a Σ-role. As answers to Σ-CQs are preserved under Σ-homomorphisms, $\mathcal{K}_1$ Σ-query entails $\mathcal{K}_2$ if there is a Σ-homomorphism from $\mathcal{M}_2$ to $\mathcal{M}_1$. However, the converse does not necessarily hold:

Example 1. Consider $\mathcal{K}_i = (\mathcal{T}_i, \{A(a)\})$, for $i = 1, 2$, where

$$\mathcal{T}_1 = \{A \sqsubseteq \exists S,\ \exists S^- \sqsubseteq \exists T,\ \exists T^- \sqsubseteq \exists S,\\ S \sqsubseteq Q,\ T \sqsubseteq Q,\ \exists Q^- \sqsubseteq \exists R\},$$

$$\mathcal{T}_2 = \{\ A \sqsubseteq \exists P,\ \exists P^- \sqsubseteq \exists R^-,\ \exists R \sqsubseteq \exists S^- \sqcap \exists Q^-,\\ \exists Q \sqsubseteq \exists Q^-,\ \exists S \sqsubseteq \exists T^-,\ \exists T \sqsubseteq \exists S^-\ \}.$$

Materialisations $\mathcal{M}_i$ of $\mathcal{K}_i$, for $i = 1, 2$, are shown in Fig. 1a. Let $\Sigma = \{Q, R, S, T\}$. Then there is no Σ-homomorphism from $\mathcal{M}_2$ to $\mathcal{M}_1$ (as $\boldsymbol{r}_\Sigma^{\mathcal{M}_2}(a, u) = \emptyset$, we can map u to, say, σ but then only the shaded part of $\mathcal{M}_2$ can be mapped Σ-homomorphically to $\mathcal{M}_1$). However, for any Σ-query $\boldsymbol{q}(\boldsymbol{x})$, $\mathcal{M}_2 \models \boldsymbol{q}(\boldsymbol{a})$ implies $\mathcal{M}_1 \models \boldsymbol{q}(\boldsymbol{a})$ as any finite subinterpretation of $\mathcal{M}_2$ can be Σ-homomorphically mapped to $\mathcal{M}_1$.

We say that $\mathcal{M}_2$ is *finitely* Σ-*homomorphically embeddable into* $\mathcal{M}_1$ if, for every *finite* subinterpretation $\mathcal{M}_2'$ of $\mathcal{M}_2$, there exists a Σ-homomorphism from $\mathcal{M}_2'$ to $\mathcal{M}_1$. Intuitively, finite $\mathcal{M}_2'$ can be regarded as a CQ whose variables are elements of $\Delta^{\mathcal{M}_2}$ and the answer variables are the $\mathsf{ind}(\mathcal{K}_2)$.

Theorem 2. *Suppose $\mathcal{K}_i$ is a consistent KB with a materialisation $\mathcal{M}_i$, $i = 1, 2$. Then $\mathcal{K}_1$ Σ-query entails $\mathcal{K}_2$ iff $\mathcal{M}_2$ is finitely Σ-homomorphically embeddable into $\mathcal{M}_1$.*

One problem with applying Theorem 2 is that materialisations are in general infinite. We address this problem by introducing finite representations of materialisations, and show that *Horn-$\mathcal{ALCHI}$* and all of its fragments enjoy having such representations. Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a KB and let $\mathcal{G} = (\Delta^\mathcal{G}, \cdot^\mathcal{G}, \leadsto)$ be a finite structure such that

– $\Delta^\mathcal{G} = \mathsf{ind}(\mathcal{K}) \cup \Omega$, for some Ω with $\mathsf{ind}(\mathcal{K}) \cap \Omega = \emptyset$,

– $(\Delta^\mathcal{G}, \cdot^\mathcal{G})$ is an interpretation with $P_i^\mathcal{G} \subseteq \mathsf{ind}(\mathcal{K}) \times \mathsf{ind}(\mathcal{K})$, for all role names P_i,

– $(\Delta^\mathcal{G}, \leadsto)$ is a directed graph (possibly containing loops) with nodes $\Delta^\mathcal{G}$ and arcs $\leadsto \subseteq \Delta^\mathcal{G} \times \Omega$, where each $u \leadsto v$ is labelled with a set $(u, v)^\mathcal{G} \neq \emptyset$ of roles satisfying the condition: if $u_1 \leadsto v$ and $u_2 \leadsto v$ then $(u_1, v)^\mathcal{G} = (u_2, v)^\mathcal{G}$.

A *path* σ in $\mathcal{G}$ is a sequence $u_0 \dots u_n$ with $u_0 \in \mathsf{ind}(\mathcal{K})$ and $u_i \leadsto u_{i+1}$ for $i < n$. Denote the last element of σ by $\mathsf{tail}(\sigma)$. The *unravelling* $\mathcal{M}$ of $\mathcal{G}$ is an interpretation $(\Delta^\mathcal{M}, \cdot^\mathcal{M})$, where $\Delta^\mathcal{M}$ is the set of paths in $\mathcal{G}$ and $\cdot^\mathcal{M}$ is given by:

$$a^\mathcal{M} = a, \qquad \text{for each } a \in \mathsf{ind}(\mathcal{K}),$$
$$A^\mathcal{M} = \{\, \sigma \mid \mathsf{tail}(\sigma) \in A^\mathcal{G} \,\},$$
$$P^\mathcal{M} = P^\mathcal{G} \cup \{\, (\sigma, \sigma u) \mid \mathsf{tail}(\sigma) \leadsto u,\ P \in (\mathsf{tail}(\sigma), u)^\mathcal{G} \,\}\\ \cup \{\, (\sigma u, \sigma) \mid \mathsf{tail}(\sigma) \leadsto u,\ P^- \in (\mathsf{tail}(\sigma), u)^\mathcal{G} \,\},$$

for concept and role names A and P. We call $\mathcal{G}$ a *generating structure for* $\mathcal{K}$ if its unravelling is a materialisation of $\mathcal{K}$. We say that a DL $\mathcal{L}$ has *finitely generated materialisations* if every $\mathcal{L}$-KB has a generating structure. In Example 1, $\mathcal{M}_2$ is generated by the structure $\mathcal{G}_2$ in Fig. 1b.

Theorem 3. *Horn-$\mathcal{ALCHI}$ and all its fragments defined above have finitely generated materialisations. Moreover,*

– *for any $\mathcal{L} \in \{\mathcal{ALCHI}, \mathcal{ALCI}, \mathcal{ALCH}, \mathcal{ALC}\}$ and any Horn-$\mathcal{L}$ KB $(\mathcal{T}, \mathcal{A})$, a generating structure can be constructed in time $|\mathcal{A}| \cdot 2^{p(|\mathcal{T}|)}$, p a polynomial*;

– *for any $\mathcal{L}$ in the $\mathcal{EL}$ and DL-Lite families introduced above and any $\mathcal{L}$-KB $(\mathcal{T}, \mathcal{A})$, a generating structure can be constructed in time $|\mathcal{A}| \cdot p(|\mathcal{T}|)$, p a polynomial.*

4 Σ-Query Entailment by Games

Suppose a DL $\mathcal{L}$ has finitely generated materialisations. We now show that the problem of checking finite Σ-homomorphic embeddability between materialisations of $\mathcal{L}$-KBs can be reduced to the problem of finding a winning strategy in a game played on the generating structures for these KBs.

For a generating structure $\mathcal{G}$ for $\mathcal{K}$ and a signature Σ, the Σ-*types* $\boldsymbol{t}_\Sigma^\mathcal{G}(u)$ and $\boldsymbol{r}_\Sigma^\mathcal{G}(u, v)$ of $u, v \in \Delta^\mathcal{G}$ are defined by:

$$\boldsymbol{t}_\Sigma^\mathcal{G}(u) = \{\, \Sigma\text{-concept name } A \mid u \in A^\mathcal{G} \,\},$$

$$\boldsymbol{r}_\Sigma^\mathcal{G}(u, v) = \begin{cases} \{\, \Sigma\text{-role } R \mid (u, v) \in R^\mathcal{G} \,\}, & \text{if } u, v \in \mathsf{ind}(\mathcal{K}),\\ \{\, \Sigma\text{-role } R \mid R \in (u, v)^\mathcal{G} \,\}, & \text{if } u \leadsto v,\\ \emptyset, & \text{otherwise,} \end{cases}$$

where $(P^-)^\mathcal{G}$ is the converse of $P^\mathcal{G}$. We write $u \leadsto^\Sigma v$ if $u \leadsto v$ and $\boldsymbol{r}_\Sigma^\mathcal{G}(u, v) \neq \emptyset$.

Suppose $\mathcal{K}_i$ is a consistent $\mathcal{L}$-KB, for $i = 1, 2$, and Σ a signature. Let $\mathcal{G}_i = (\Delta^{\mathcal{G}_i}, \cdot^{\mathcal{G}_i}, \leadsto_i)$ be a generating structure for $\mathcal{K}_i$ and let $\mathcal{M}_i$ be its unravelling; $\mathcal{G}_i^\Sigma$ and $\mathcal{M}_i^\Sigma$ denote the restrictions of $\mathcal{G}_i$ and $\mathcal{M}_i$ to Σ. We begin with a very simple game on the finite generating structure $\mathcal{G}_2^\Sigma$ and the possibly infinite materialisation $\mathcal{M}_1^\Sigma$.

Infinite Game. This game is played by two players. Intuitively, player 1 tries to construct a homomorphism, while player 2 wants to impede him by choosing a path in $\mathcal{M}_2$ to which player 1 cannot find a homomorphic image given his previous choices. The *states* of the game are of the form $\mathfrak{s}_i = (u_i \mapsto \sigma_i)$, for $i \geq 0$, where $u_i \in \Delta^{\mathcal{G}_2}$ and $\sigma_i \in \Delta^{\mathcal{M}_1}$ satisfy the following condition:

(s$_1$) $\boldsymbol{t}_\Sigma^{\mathcal{G}_2}(u_i) \subseteq \boldsymbol{t}_\Sigma^{\mathcal{M}_1}(\sigma_i)$.

The game starts in a state $\mathfrak{s}_0 = (u_0 \mapsto \sigma_0)$ with $\sigma_0 = u_0$ in case $u_0 \in \mathsf{ind}(\mathcal{K}_2)$ belongs to a Σ-concept or a Σ-role. In each round $i > 0$, player 2 challenges player 1 with some $u_i \in \Delta^{\mathcal{G}_2}$ such that $u_{i-1} \rightsquigarrow_2^\Sigma u_i$. Player 1 has to respond with a $\sigma_i \in \Delta^{\mathcal{M}_1}$ satisfying **(s$_1$)** and

(s$_2$) $\boldsymbol{r}_\Sigma^{\mathcal{G}_2}(u_{i-1}, u_i) \subseteq \boldsymbol{r}_\Sigma^{\mathcal{M}_1}(\sigma_{i-1}, \sigma_i)$.

This gives the next state $\mathfrak{s}_i = (u_i \mapsto \sigma_i)$. Note that of all the u_i only u_0 may be an ABox individual; however, there is no such a restriction on the σ_i. A *play of length* $n \geq 0$ *starting from* $\mathfrak{s}_0$ is any sequence $\mathfrak{s}_0, \dots, \mathfrak{s}_n$ of states obtained as described above. For an ordinal $\lambda \leq \omega$, we say that player 1 has a λ*-winning strategy in the game* $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ *starting from a state* $\mathfrak{s}_0$ if, for any play of length $i < \lambda$, which starts from $\mathfrak{s}_0$ and conforms with this strategy, and any challenge of player 2 in round $i+1$, player 1 has a response. The next theorem gives a game-theoretic flavour to the criterion of Theorem 2.

Theorem 4. *$\mathcal{M}_2$ is finitely Σ-homomorphically embeddable into $\mathcal{M}_1$ iff the following conditions hold*:

(abox) $\boldsymbol{t}_\Sigma^{\mathcal{M}_2}(a) \subseteq \boldsymbol{t}_\Sigma^{\mathcal{M}_1}(a)$, $\boldsymbol{r}_\Sigma^{\mathcal{M}_2}(a,b) \subseteq \boldsymbol{r}_\Sigma^{\mathcal{M}_1}(a,b)$, *for any* $a, b \in \mathsf{ind}(\mathcal{K}_2)$ *that belong to a Σ-concept or a Σ-role*;

(win) *for any $u_0 \in \Delta^{\mathcal{G}_2}$ and $n < \omega$, there exists $\sigma_0 \in \Delta^{\mathcal{M}_1}$ such that player 1 has an n-winning strategy in the game $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ starting from $(u_0 \mapsto \sigma_0)$.*

Example 2. Consider $\mathcal{G}_2^\Sigma$ for $\mathcal{K}_2$, $\mathcal{M}_1^\Sigma$ for $\mathcal{K}_1$ and Σ from Example 1. For each $n < \omega$, player 1 has an n-winning strategy from $(u \mapsto \sigma_0^n)$: for $n = 4$, it is shown in Fig. 1b by dotted lines (in round 2, player 2 has two possible challenges).

The criterion of Theorem 4 does not seem to be a big improvement on Theorem 2 as we still have to deal with an infinite materialisation. Our aim now is to replace **(win)** by a more complex game on the *finite* generating structures $\mathcal{G}_2$ and $\mathcal{G}_1$. We consider four types, τ, of strategies in $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ and for each of them we define a game $G_\Sigma^\tau(\mathcal{G}_2, \mathcal{G}_1)$ such that, for any $u_0 \in \Delta^{\mathcal{G}_2}$, the following conditions are equivalent:

($<\boldsymbol{\omega}^\tau$) for every $n < \omega$, player 1 has an n-winning τ-strategy in $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ starting from some $(u_0 \mapsto \sigma_0^n)$;

($\boldsymbol{\omega}^\tau$) player 1 has an ω-winning strategy in $G_\Sigma^\tau(\mathcal{G}_2, \mathcal{G}_1)$ starting from some state depending on u_0 and τ.

We begin with simplest 'forward' winning strategies.

Forward Strategies. We say that a λ-strategy ($\lambda \leq \omega$) for player 1 in the game $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ is *forward* if, for any play of length $i - 1 < \lambda$, which conforms with this strategy, and any challenge $u_{i-1} \rightsquigarrow_2^\Sigma u_i$ by player 2, the response σ_i of player 1 is such that either $\sigma_{i-1}, \sigma_i \in \mathsf{ind}(\mathcal{K}_1)$ or $\sigma_i = \sigma_{i-1}w$, for some $w \in \Delta^{\mathcal{G}_1}$. If the $\mathcal{G}_i$, $i = 1, 2$, are such that the Σ-labels on $\rightsquigarrow_i$-edges contain no inverse roles, then *every* strategy in $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ is forward. This is the case for DLs without inverse roles: *Horn*-$\mathcal{ALCH}$, $\mathcal{EL}$, etc.

The existence of a forward λ-winning strategy for player 1 in $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ is equivalent to the existence of such a strategy in the *game* $G_\Sigma^f(\mathcal{G}_2, \mathcal{G}_1)$, which is defined similarly to $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ except that it is played on $\mathcal{G}_2$ and $\mathcal{G}_1$, and the response $w_i \in \Delta^{\mathcal{G}_1}$ of player 1 to a challenge $u_{i-1} \rightsquigarrow_2^\Sigma u_i$ must be such that either $w_{i-1}, w_i \in \mathsf{ind}(\mathcal{K}_1)$ or $w_{i-1} \rightsquigarrow_1 w_i$. This game is a standard reachability game on finite graphs, where the existence of ω-winning strategies for player 1 can be checked in polynomial time in the size of $\mathcal{G}_1$ and $\mathcal{G}_2$ [Mazala, 2001]. By Theorem 3, we obtain the P and ExpTime upper complexity bounds for $\mathcal{ELH}$ and *Horn*-$\mathcal{ALCH}$, respectively. In contrast to forward strategies, the winning strategies of Example 2 can be described as 'backward.'

Backward Strategies. A λ-strategy for player 1 in $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ is *backward* if, for any play of length $i - 1 < \lambda$, which conforms with this strategy, and any challenge $u_{i-1} \rightsquigarrow_2^\Sigma u_i$ by player 2, the response σ_i of player 1 is the *immediate predecessor* of σ_{i-1} in $\mathcal{M}_1$ in the sense that $\sigma_{i-1} = \sigma_i w$, for some $w \in \Delta^{\mathcal{G}_1}$; player 1 loses if $\sigma_{i-1} \in \mathsf{ind}(\mathcal{K}_1)$. Note that, since $\mathcal{M}_1$ is tree-shaped, the response of player 1 to any other challenge $u_{i-1} \rightsquigarrow_2^\Sigma u_i'$ must be the same σ_i. That is why the states of the game $G_\Sigma^b(\mathcal{G}_2, \mathcal{G}_1)$ are of the form $(\Xi_i \mapsto w_i)$, where Ξ_i is the *set* of all $\rightsquigarrow_2^\Sigma$-successors of elements of Ξ_{i-1} (forward strategies need only one successor). The more complex structure of the states leads to an increase in the complexity: checking whether player 1 has an ω-winning strategy in $G_\Sigma^b(\mathcal{G}_2, \mathcal{G}_1)$ is coNP-hard.

Observe that in the case of *DL-Lite$_{core}$* and *DL-Lite$_{horn}$* (which have inverse roles but no RIs), generating structures $\mathcal{G} = (\Delta^\mathcal{G}, \cdot^\mathcal{G}, \rightsquigarrow)$ are so that, for any $u \in \Delta^\mathcal{G}$ and R, there is *at most one* v with $u \rightsquigarrow v$ and $R \in \boldsymbol{r}^\mathcal{G}(u, v)$ [Kontchakov *et al.*, 2010a]. As a result, any n-winning strategy consists of a (possibly empty) backward part followed by a (possibly empty) forward part. Moreover, in the backward games for these DLs, the sets Ξ_i are *singletons*. Thus, the number of states in the combined backward/forward games is polynomial, and the existence of winning strategies is in P.

In general, however, the forward strategy in the combination is not enough, and we require start-bounded strategies.

Start-Bounded Strategies. A strategy for player 1 in the game $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ starting from $(u_0 \mapsto \sigma_0)$ is *start-bounded* if it never leads to $(u_i \mapsto \sigma_i)$ with $\sigma_0 = \sigma_i w$, for some $w \in \Delta^{\mathcal{G}_1}$ and $i > 0$. In other words, player 1 cannot use elements of $\mathcal{M}_1$ that are located closer to the ABox than σ_0; the ABox individuals in $\mathcal{M}_1$ can only be used if $\sigma_0 \in \mathsf{ind}(\mathcal{K}_1)$. Consider $\mathcal{G}_2^\Sigma$ and $\mathcal{M}_1^\Sigma$ in Fig. 2a. Player 1 has a winning start-bounded strategy from $(u_2 \mapsto \sigma_1)$ as shown in Fig. 2a by dashed lines (indices indicate rounds). Observe that player 1 moves forwards and backwards along $\rightsquigarrow_1$ in $\mathcal{M}_1$: $\sigma_4 = \sigma_3 w$ is visited in round 2 between visits to σ_3 in rounds 1 and 3.

Start-bounded finite game $G_\Sigma^s(\mathcal{G}_2, \mathcal{G}_1)$ ensures that player 1 moves only forwards along $\rightsquigarrow_1$ in $\mathcal{G}_1$ and so, he has to guess *all* elements of $\mathcal{G}_2$ that are mapped to the same element in $\mathcal{M}_1$. The states of $G_\Sigma^s(\mathcal{G}_2, \mathcal{G}_1)$ are of the form

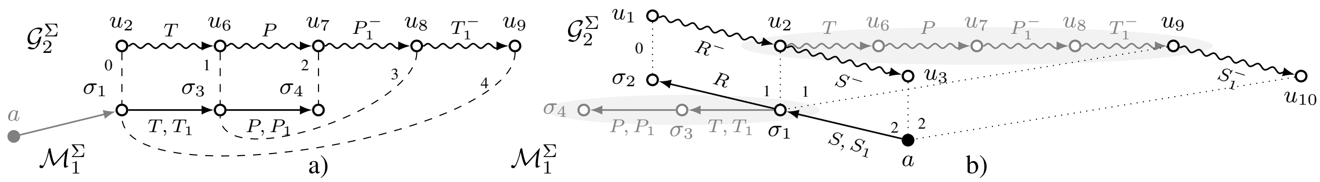


Figure 2: a) start-bounded strategy and b) general strategy as a composition of a backward and start-bounded strategies.

$\mathfrak{s}_i = (\Gamma_i, \Xi_i \mapsto w_i)$, where Ξ_i is the guess and Γ_i is required to ensure that only forward moves are possible.

Consider $\mathcal{G}_2^\Sigma$ and $\mathcal{M}_1^\Sigma$ in Fig. 2a. Suppose that $\mathcal{G}_1$ is isomorphic to $\mathcal{M}_1$ and denote by w_i the element of $\mathcal{G}_1$ that corresponds to σ_i in $\mathcal{M}_1$. Then player 1 has an ω-winning strategy in $G_\Sigma^s(\mathcal{G}_2, \mathcal{G}_1)$ from $(\emptyset, \{u_2, u_9\} \mapsto w_1)$. Player 2 challenges with $u_2 \rightsquigarrow_2^\Sigma u_6$, and player 1 responds with $(\{u_2, u_9\}, \{u_6, u_8\} \mapsto w_3)$. Then player 2 picks $u_6 \rightsquigarrow_2^\Sigma u_7$ and player 1 responds with $(\{u_6, u_8\}, \{u_7\} \mapsto w_4)$, where the game ends. Note the crucial guesses $\{u_2, u_9\} \mapsto w_1$ and $\{u_6, u_8\} \mapsto w_3$ made by player 1. If player 1 responded with $(\{u_2, u_9\}, \{u_6\} \mapsto w_3)$ (and failed to guess that u_8 must also be mapped to w_3), then after the challenge $u_6 \rightsquigarrow_2^\Sigma u_7$ and the only possible response $(\{u_6\}, \{u_7\} \mapsto w_4)$), player 2 would pick $u_7 \rightsquigarrow_2^\Sigma u_8$, to which player 1 could not respond.

General Strategies. A general winning strategy in the game $G_\Sigma(\mathcal{G}_2, \mathcal{M}_1)$ is composed of one backward and a number of start-bounded strategies. Consider, for example, $\mathcal{G}_2^\Sigma$ and $\mathcal{M}_1^\Sigma$ in Fig. 2b. Starting from $(u_1 \mapsto \sigma_2)$, player 1 can respond to the challenges $u_1 \rightsquigarrow_2^\Sigma u_2 \rightsquigarrow_2^\Sigma u_3$ according to the backward strategy; to $u_2 \rightsquigarrow_2^\Sigma u_6 \rightsquigarrow_2^\Sigma u_7 \rightsquigarrow_2^\Sigma u_8 \rightsquigarrow_2^\Sigma u_9$ according to the start-bounded strategy as above, see Fig. 2a; while $u_9 \rightsquigarrow_2^\Sigma u_{10}$ is again responded according to the backward strategy. We can combine the two backward strategies into a single one, but keep the start-bounded one separate.

In general, the finite game $G_\Sigma^g(\mathcal{G}_2, \mathcal{G}_1)$ begins as the backward game but with states of the form $(\Xi_i \mapsto w_i, \Psi_i)$, where Ξ_i and w_i are as above and Ψ_i indicates initial challenges in start-bounded games. In each round $i > 0$, player 2 can choose from two options. First, if $w_{i-1} \notin \mathsf{ind}(\mathcal{K}_1)$, he can challenge player 1 with the set Ψ_{i-1} (that is, similar to the backward game but with a possibly smaller Ψ_{i-1} instead of the set of all successors). Second, player 2 can launch the start-bounded game from $(\emptyset, \Xi_{i-1} \mapsto w_{i-1})$, where his first challenge cannot be picked from Ψ_{i-1}.

The full game graph is exponential in the size of the generating structures, and so checking whether player 1 has an ω-winning strategy can also be done in exponential time. This matches the ExpTime-hardness of checking whether player 1 has an ω-winning strategy in the *start-bounded* game.

Theorem 5. *For combined complexity, both KB Σ-query inseparability and KB Σ-query entailment are*

(f) P-*complete in $\mathcal{EL}$ and $\mathcal{ELH}$ and* ExpTime-*complete in Horn-$\mathcal{ALC}$ and Horn-$\mathcal{ALCH}$*;

(b+f) P-*complete in DL-Lite$_{core}$ and DL-Lite$_{horn}$*;

(b+s) ExpTime-*complete in DL-Lite$^{\mathcal{H}}_{horn}$ and DL-Lite$^{\mathcal{H}}_{core}$*; 2ExpTime-*complete in Horn-$\mathcal{ALCHI}$, Horn-$\mathcal{ALCI}$.*

For data complexity, all these problems are P-*complete.*

5 Related Work

Σ-query inseparability of KBs has not been investigated systematically before. However, the polynomial upper bound for $\mathcal{EL}$ was established as a preliminary step to study TBox query inseparability [Lutz and Wolter, 2010]. This notion was also used to study forgetting in *DL-Lite$^{\mathcal{N}}_{bool}$* [Wang *et al.*, 2010].

Σ-query inseparability of KBs is closely related to knowledge exchange between KBs and inseparability between TBoxes. Suppose $\mathcal{K}_1$ and $\mathcal{K}_2$ are KBs given in disjoint signatures Σ_1 and Σ_2, and $\mathcal{T}_{12}$ consists of inclusions of the form $S_1 \sqsubseteq S_2$, where the S_i are concept (or role) names in Σ_i. Then deciding whether $\mathcal{K}_1 \cup \mathcal{T}_{12} \equiv_{\Sigma_2} \mathcal{K}_2$ is called the *membership problem for universal CQ-solutions*. For DLs $\mathcal{L}$ with role inclusions, this problem is in fact a Σ_2-query inseparability problem in $\mathcal{L}$, and so the complexity upper bounds for Σ-query inseparability can be applied directly to obtain upper bounds for the membership problem. Conversely, Σ-query entailment for any of our DLs $\mathcal{L}$ is LogSpace-reducible to the membership problem for universal CQ-solutions in $\mathcal{L}$, and so we can also apply complexity lower bounds for query inseparability of KBs.

As for TBox inseparability, recall that $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-query inseparable if, for *all* Σ-ABoxes $\mathcal{A}$, the KBs $(\mathcal{T}_1, \mathcal{A})$ and $(\mathcal{T}_2, \mathcal{A})$ are Σ-query inseparable. This notion has been extensively studied [Kontchakov *et al.*, 2010b; Lutz and Wolter, 2010; Konev *et al.*, 2011; 2012]. TBox and KB inseparabilities have different applications. The former supports ontology engineering when data is unknown or changes frequently: one can equivalently replace one TBox with another only if they return the same answers to queries over *every* Σ-ABox. In contrast, KB inseparability is useful in applications where data is stable—such as knowledge exchange or variants of module extraction and forgetting with fixed data—in order to use the KB in a new application or as a compilation step to make CQ answering more efficient. For many DLs, TBox Σ-query inseparability is harder than KB query inseparability: in *DL-Lite$_{horn}$*, the space of relevant ABox counterexamples is exponential and, in fact, TBox inseparability is NP-hard, while KB inseparability is in P. Similarly, Σ-query inseparability of $\mathcal{EL}$ KBs is tractable, while Σ-query inseparability of TBoxes is ExpTime-complete. The complexity of TBox inseparability for Horn-DLs extending Horn-$\mathcal{ALC}$ is still unknown. For work on other notions of TBox inseparability and the corresponding notions of modules and forgetting, the reader is referred to [Cuenca Grau *et al.*, 2008; Konev *et al.*, 2009; Del Vescovo *et al.*, 2011; Nikitina and Rudolph, 2014; Nikitina and Glimm, 2012; Lutz *et al.*, 2012; Koopmann and Schmidt, 2014a; Nortje *et al.*, 2013].

References

[Artale *et al.*, 2009] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *JAIR*, 39:1–69, 2009.

[Arenas *et al.*, 2012] M. Arenas, E. Botoeva, D. Calvanese, V. Ryzhikov, and E. Sherkhonov. Exchanging description logic knowledge bases. In *KR*, 2012.

[Arenas *et al.*, 2013] M. Arenas, J. Pérez, and J. L. Reutter. Data exchange beyond complete data. *JACM*, 60:28, 2013.

[Arenas *et al.*, 2014] M. Arenas, P. Barceló, L. Libkin, and F. Murlak. *Foundations of Data Exchange*. CUP, 2014.

[Baader *et al.*, 2003] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. CUP, 2003 (2nd edition, 2007).

[Baader *et al.*, 2005] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *IJCAI*, 2005.

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

[Cuenca Grau and Motik, 2012] B. Cuenca Grau and B. Motik. Reasoning over ontologies with hidden content: The import-by-query approach. *JAIR*, 45:197–255, 2012.

[Cuenca Grau *et al.*, 2008] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *JAIR*, 31:273–318, 2008.

[Del Vescovo *et al.*, 2011] C. Del Vescovo, B. Parsia, U. Sattler, and T. Schneider. The modular structure of an ontology: Atomic decomposition. In *IJCAI*, 2011.

[Eiter *et al.*, 2008] T. Eiter, G. Gottlob, M. Ortiz, and M. Simkus. Query answering in the description logic Horn-$\mathcal{SHIQ}$. In *JELIA*, 2008.

[Giese *et al.*, 2013] M. Giese, D. Calvanese, P. Haase, I. Horrocks, Y. Ioannidis, H. Kllapi, M. Koubarakis, M. Lenzerini, R. Möller, M. Rodriguez-Muro, Ö. Özcep, R. Rosati, R. Schlatte, M. Schmidt, A. Soylu, A. Waaler. Scalable end-user access to big data. In *Big Data Computing*, 2013.

[Hitzler *et al.*, 2009] P. Hitzler, M. Krötzsch, S. Rudolph. *Foundations of Semantic Web Technologies*. CRC, 2009.

[Hustadt *et al.*, 2005] U. Hustadt, B. Motik, and U. Sattler. Data complexity of reasoning in very expressive description logics. In *IJCAI*, 2005.

[Jiménez-Ruiz *et al.*, 2011] E. Jiménez-Ruiz, B. Cuenca Grau, I. Horrocks, and R. Berlanga Llavori. Supporting concurrent ontology development: Framework, algorithms and tool. *Data Knowl. Eng.*, 70(1):146–164, 2011.

[Kazakov, 2009] Y. Kazakov. Consequence-driven reasoning for Horn-$\mathcal{SHIQ}$ ontologies. In *IJCAI*, 2009.

[Konev *et al.*, 2009] B. Konev, D. Walther, and F. Wolter. Forgetting and uniform interpolation in large-scale description logic terminologies. In *IJCAI*, 2009.

[Konev *et al.*, 2011] B. Konev, R. Kontchakov, M. Ludwig, T. Schneider, F. Wolter, M. Zakharyaschev. Conjunctive query inseparability of OWL 2 QL TBoxes. In *AAAI*, 2011.

[Konev *et al.*, 2012] B. Konev, M. Ludwig, D. Walther, and F. Wolter. The logical difference for the lightweight description logic $\mathcal{EL}$. *JAIR*, 44:633–708, 2012.

[Kontchakov and Zakharyaschev, 2014] R. Kontchakov and M. Zakharyaschev. An introduction to description logics and query rewriting. In *RW*, 2014.

[Kontchakov *et al.*, 2010a] R. Kontchakov, C. Lutz, D. Toman, F. Wolter, and M. Zakharyaschev. The combined approach to query answering in DL-Lite. In *KR*, 2010.

[Kontchakov *et al.*, 2010b] R. Kontchakov, F. Wolter, and M. Zakharyaschev. Logic-based ontology comparison and module extraction, with an application to DL-Lite. *Artif. Intell.*, 174:1093–1141, 2010.

[Koopmann and Schmidt, 2014a] P. Koopmann and R. A. Schmidt. Count and forget: Uniform interpolation of $\mathcal{SHQ}$-ontologies. In *IJCAR*, 2014.

[Koopmann and Schmidt, 2014b] P. Koopmann and R. A. Schmidt. Forgetting and uniform interpolation for $\mathcal{ALC}$-ontologies with ABoxes. In *DL*, 2014.

[Krötzsch *et al.*, 2013] M. Krötzsch, S. Rudolph, and P. Hitzler. Complexities of Horn description logics. *ACM Trans. Comput. Log.*, 14(1):2, 2013.

[Lutz and Wolter, 2010] C. Lutz and F. Wolter. Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. *J. Symbolic Computation*, 45(2):194–228, 2010.

[Lutz *et al.*, 2009] C. Lutz, D. Toman, and F. Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *IJCAI*, 2009.

[Lutz *et al.*, 2012] C. Lutz, I. Seylan, and F. Wolter. An automata-theoretic approach to uniform interpolation and approximation in the description logic $\mathcal{EL}$. In *KR*, 2012.

[Mazala, 2001] R. Mazala. Infinite games. In *Automata, Logics, and Infinite Games*, pages 23–42, 2001.

[Nikitina and Glimm, 2012] N. Nikitina and B. Glimm. Hitting the sweetspot: Economic rewriting of knowledge bases. In *ISWC*, 2012.

[Nikitina and Rudolph, 2014] N. Nikitina and S. Rudolph. (Non-)Succinctness of uniform interpolants of general terminologies in the description logic $\mathcal{EL}$. *Artif. Intell.*, 215:120–140, 2014.

[Nortje *et al.*, 2013] R. Nortje, K. Britz, and T. Meyer. Reachability modules for the description logic $\mathcal{SRIQ}$. In *LPAR*, 2013.

[Poggi *et al.*, 2008] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. Data Semantics*, X:133–173, 2008.

[Polleres *et al.*, 2013] A. Polleres, A. Hogan, R. Delbru, and J. Umbrich. RDFS and OWL reasoning for Linked Data. In *RW*, 2013.

[Rosati, 2007] R. Rosati. On conjunctive query answering in EL. In *DL*, 2007.

[Shvaiko and Euzenat, 2013] P. Shvaiko and J. Euzenat. Ontology matching: State of the art and future challenges. *IEEE Trans. Knowl. Data Eng.*, 25(1):158–176, 2013.

[Stuckenschmidt *et al.*, 2009] H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*. Springer, 2009.

[Wang *et al.*, 2010] Z. Wang, K. Wang, R. W. Topor, and J. Z. Pan. Forgetting for knowledge bases in DL-Lite. *Ann. Math. Artif. Intell.*, 58(1–2):117–151, 2010.