Using Ontology-Based Data Access to Enable Context Recognition in the Presence of Incomplete Information

Dissertation

zur Erlangung des akademischen Grades
Doktoringenieur (Dr.-Ing.)

vorgelegt an der
Technischen Universität Dresden
Fakultät Informatik

eingereicht von
Dipl.-Inf. Veronika Thost
geboren am 16. Juni 1988 in Konstanz

verteidigt am 19. Juni 2017

Gutachter:
Prof. Dr.-Ing. Franz Baader, Technische Universität Dresden
Prof. Dr. Carsten Lutz, Universität Bremen

Dresden, im August 2017

Abstract

Ontology-based data access (OBDA) augments classical query answering in databases by including domain knowledge provided by an ontology. An ontology captures the terminology of an application domain and describes domain knowledge in a machine-processable way. Formal ontology languages additionally provide semantics to these specifications. Systems for OBDA thus may apply logical reasoning to answer queries; they use the ontological knowledge to infer new information, which is only implicitly given in the data. Moreover, they usually employ the open-world assumption, which means that knowledge not stated explicitly in the data or inferred is neither assumed to be true nor false. Classical OBDA regards the knowledge however only w.r.t. a single moment, which means that information about time is not used for reasoning and hence lost; in particular, the queries generally cannot express temporal aspects.

We investigate temporal query languages that allow to access temporal data through classical ontologies. In particular, we study the computational complexity of temporal query answering regarding ontologies written in lightweight description logics, which are known to allow for efficient reasoning in the atemporal setting and are successfully applied in practice. Furthermore, we present a so-called rewritability result for ontology-based temporal query answering, which suggests ways for implementation. Our results may thus guide the choice of a query language for temporal OBDA in data-intensive applications that require fast processing, such as context recognition.

Acknowledgments

First of all, I would like to thank my supervisor Franz Baader for all his advice and for giving me the chance to work in his group. Likewise, I want to thank my co-author Stefan for his continuous support and advice, and for answering myriads of questions in a remarkably calm fashion. I am deeply grateful to my parents for supporting me whenever I need it. I would like to thank my friends for all their understanding when I repeatedly excelled by absence. I want to thank many people from the Chair for Automata Theory. In particular, I would like to thank Anni and Marcel for supporting me a lot, especially in my first year. I thank the fellow HAEC colleagues I collaborated with over time. I want to thank Kerstin for all her emotional support and for caring about everything. I would like to thank Markus for his advice and also all others of the Knowledge-Based Systems Group for many interesting discussions. Especially, I thank Ana for her support in the final months. I also want to thank Carsten Lutz for sharing his ideas and discussing our work.

This work was financially supported by the Deutsche Forschungsgemeinschaft in the Collaborative Research Center 912 (HAEC).

Contents

List of Figures

1 Introduction

Ontologies play an important role as a semantic layer for data access in various areas such as the Semantic Web [BHL01], medicine [Rec+94; SCC97], and enterprise applications [Bus03; Ara+08; Kha+15]. They capture the terminology of an application domain and describe domain knowledge in a machine-processable way. Formal ontology languages additionally provide semantics to these specifications. In contrast to standard database systems, systems for *ontology-based data access* (OBDA) thus may apply *logical reasoning* to answer queries; they use the ontological knowledge to infer new information, which is only implicitly given in the data. Moreover, they usually employ the *open-world assumption*, which means that knowledge not stated explicitly in the data or inferred is neither assumed to be true nor false. This faithfully models the real world and also differs from database query answering, which assumes knowledge not present in the data to be false.

All these features make ontologies valuable tools for systems that integrate heterogeneous data sources and need to automatically interpret the data, to support data analysis or to fully-automatedly recognize complex contexts; also multi-agent systems profit from the semantic interoperability. This has been generally recognized and several standardized ontologies have recently been published, especially for domains where heterogeneous data sources are usual or different agents have to communicate seamlessly, such as for sensor networks [Com+12] and robotics and automation [Ont15]. Often, the processed data is changing and thus temporal in that it is associated to specific points in time, and this temporal dimension is critical for analysis or for describing and recognizing real-world contexts. Sensors, for example, produce *streams of data*. Classical ontology-based data access regards the knowledge however only w.r.t. a single moment, which means that information about time is not used for reasoning and thus lost; in particular, the queries generally cannot express temporal aspects.

This work therefore investigates temporal query languages that allow to access temporal data through classical ontologies. In particular, we study the computational complexity of temporal query answering regarding ontologies written in *lightweight description logics*, which are known to allow for efficient reasoning in the atemporal setting and are successfully applied in practice [Rec+94; SCC97; Kha+15]. Our results may thus guide the choice of a query language for temporal OBDA in data-intensive applications that require fast processing.

This chapter provides a rather informal introduction to the topic. In Section 1.1, we generally describe OBDA. We then introduce description logics and temporal query answering in Sections 1.2 and 1.3, respectively. In Section 1.4, we specify our contributions and give an overview of the following chapters.

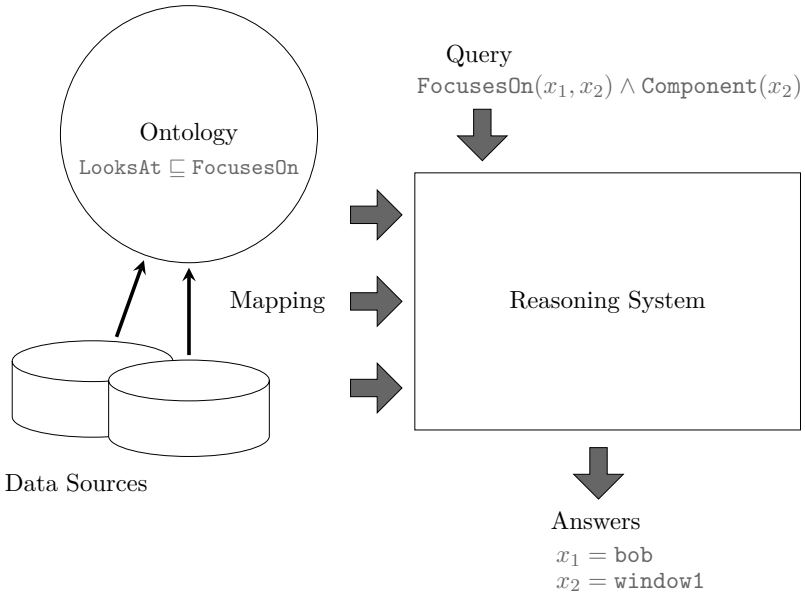

Figure 1.1: Architecture of ontology-based data access. If the ontological axiom `LooksAt` $\sqsubseteq$ `FocusesOn` ("every tuple in the relation `LooksAt` is also in the relation `FocusesOn`") is taken into account for answering the example query over the data sources (see Figure 1.2), then a reasoning system outputs the answers depicted.

1.1 Ontology-Based Data Access

Today, many applications need to process large amounts of heterogeneous *data* growing over time—the famous "big data". *Data integration* is critical for managing and analyzing such information and demands a common, well-defined vocabulary. Otherwise, misinterpretation may lead to lacking or even wrong consequences.

Ontologies play a fundamental role in this context. In computer science, an ontology can be described as in [Ont15, Introduction]:

> *"It formally specifies the key concepts, properties, relationships, and axioms of a given domain. Unlike taxonomies, which provide only a set of vocabulary and a single type of relationship between terms, an ontology provides a richer set of relationships, constraints, and rules. In general, ontologies make the relevant knowledge about a domain explicit in a computer-interpretable format, allowing software to reason over that knowledge to infer new information."*

In summary, ontologies are formal domain models that provide semantic interoperability and additionally allow for knowledge inference.

The general architecture of ontology-based data access is illustrated in Figure 1.1. As it is common in data integration, the original data sources are mapped to a global schema—here represented by the ontology—that integrates the sources and allows to access the data using a shared vocabulary while the peculiarities of the sources stay

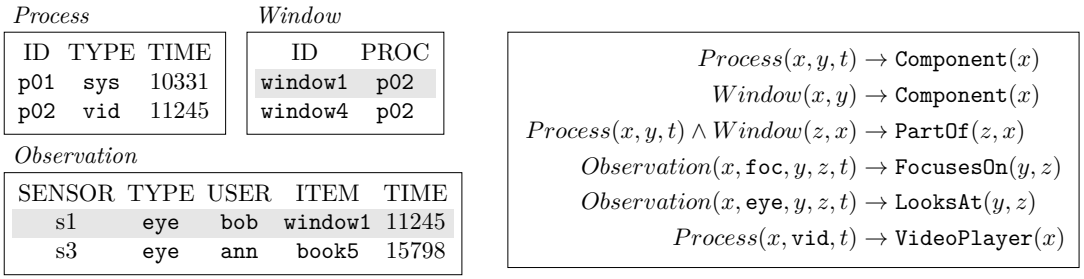

Figure 1.2: Data sources and mapping for the example in Figure 1.1; the variables in the mapping are universally quantified.

transparent [Len02]; for example, observations of different types of sensors monitoring eye movement (`eye`) or human focus (`foc`) may be mapped to corresponding ontological relations such as `LooksAt` and `FocusesOn`. Example sources and the so-called *global-as-view* mapping (partly) are depicted in more detail in Figure 1.2. The two first mappings map both relations *Process* and *Window* to the ontological concept `Component` and hence show how distinct sources can be integrated easily. Note that the mappings can be considered as ontological statements as well. Yet, an ontology may also contain constraints to detect possible inconsistencies in the data and inference rules to derive additional knowledge. For example, it may contain a rule as depicted in Figure 1.1 (in description logic notation), stating that someone looking at something focuses on it; this is useful if the system, for some reason, did not receive data from a sensor of type `foc`, directly capturing the focus. Applications of ontologies for context recognition are also described in [Dar+13; Häh+14]. If the data then is queried through a reasoning system as depicted (i.e., instead of a traditional database system), the ontological knowledge is taken into account by the logical reasoning applied during the answering process. This is commonly referred to as *ontology-based data access* or *ontology-based query answering*[1].

The global schema often only represents a view of the data, which means that the corresponding database only exists virtually and, in reality, the data is left in the original sources (i.e., instead of materializing it according to the global schema). For answering queries over the global schema—also if a traditional database schema instead of an ontology is regarded—, the queries then have to be reformulated in terms of the source schemas. This is known as *query rewriting* and has been extensively studied for *conjunctive queries* (CQs) [Len02]. A CQ is a conjunction of first-order atoms where the variables may be existentially quantified and the remaining variables represent the answers to the query. For example, the following CQ $\text{Context}_{\text{Focus}}$ can be used to recognize a complex context, by retrieving all those components x_1 and users x_2 such that x_1 is a subcomponent of some component y_1 which has a part y_2 the user focuses on:

$$\begin{aligned}\exists y_1, y_2.\texttt{Component}(x_1) \wedge \texttt{Component}(y_1) \wedge \texttt{Component}(y_2) \wedge \\ \texttt{PartOf}(x_1, y_1) \wedge \texttt{PartOf}(y_2, y_1) \wedge \texttt{FocusesOn}(x_2, y_2).\end{aligned} \tag{1.1}$$

[1] In the following, we usually drop the prefix ontology-based and simply refer to "query answering". If an ontology is not considered, we use the notion "database query answering".

Regarding the sources from Figure 1.2, query rewriting then creates a disjunction of conjunctive queries that together represent all possibilities of incorporating the mapping information into the CQ $\text{Context}_{\text{Focus}}$:

$$
\begin{aligned}
&\Big(\exists y_1, y_2, u_1, \ldots, u_4, v_1, \ldots, v_4. \\
&\qquad Process(x_1, u_1, v_1) \wedge Process(y_1, u_2, v_2) \wedge Process(y_2, u_3, v_3) \wedge \\
&\qquad Window(x_1, y_1) \wedge Window(y_2, y_1) \wedge Observation(u_4, \texttt{foc}, x_2, y_2, v_4)\Big) \vee \\
&\Big(\exists y_1, y_2, u_1, u_2, v_1, v_2. \\
&\qquad Window(x_1, y_1) \wedge Window(y_2, y_1) \wedge Process(y_2, u_1, v_1) \wedge \\
&\qquad Observation(u_2, \texttt{foc}, x_2, y_2, v_2)\Big) \vee \\
&\quad \ldots
\end{aligned}
\tag{1.2}
$$

Note that the rewritten query, called *rewriting*, may contain CQs that will never retrieve answers, such as the first one, where x_1 is considered to be both a process and a window; in practice such CQs could be dropped for optimization.

Both aspects of ontological modeling, the formality and the possibility for logical inferencing, are long-standing areas of research in computer science. The importance of formal modeling was recognized early and there are well-established techniques for all kinds of use cases, such as entity-relationship modeling [Che76] for representing database schemas and UML [OMG15] for hard and software artifacts. The search for logics that provide sufficient expressive power and, at the same time, allow for efficient inferencing—alike human reasoning—is performed in the field of artificial intelligence. The usually required efficiency makes the design of such logics challenging and is the reason for the restricted expressiveness of many formalisms. For that reason, the logics are often tailored to certain use cases, and ever new use cases make it an ongoing research.

Also in the area of databases, ontological axioms have been considered since the early 1970s; for instance, global-as-view mappings and key constraints, which specify columns in a relation that uniquely identify a tuple (see [AHV95] for a general overview of database constraints). Many are standard in database management systems today. However, observe that they primarily target technical issues, such as data coherence and integration, rather than an extensive description of domain knowledge.

Particularly user-friendly approaches of ontological knowledge representation emerged with semantic networks [Qui67] and frames [Min74], which allow for modeling in patterns so as humans are thinking; for example, both discern concepts and relations between them, such as is-a and property relations. Reasoning could thus be described in a declarative fashion. Yet, because of a lack of formal semantics, the actual "reasoning" in systems was based on ad-hoc procedures. This deficiency lead to the development of *description logics* in the mid-1980s, which follow a similar modeling paradigm but provide formal semantics (see [Baa+07] for a more detailed description of this development). Since then, description logics have been studied extensively, and they also represent the logical background of the most prominent ontology language today, the *Web Ontology Language* OWL, a W3C standard [DS04].

1.2 Lightweight Description Logics as Ontology Languages

Description logics (DLs) are a family of logical formalisms that were originally designed for terminological modeling and *decidable* reasoning, while featuring both sufficient expressivity and readability [Baa+07]. Over time, several further use cases have come to the fore and new DLs tailored to those have been developed. Today, the family comprises *lightweight* DLs, such as $\mathcal{EL}$ [BKM99] and many *DL-Lite* logics [Cal+07b; Art+09], which allow for *tractable* reasoning (i.e., reasoning in polynomial time); the prototypical DL $\mathcal{ALC}$ [SS91], which is minimal propositionally complete[2]; and very expressive DLs such as $\mathcal{SROIQ}$ [HKS06], which represents the basis of OWL 2[3]. Note that *DL-Lite* was originally proposed as a particular DL [Cal+04]; the term today however usually refers to a family of DLs comprising logics that have been developed subsequently and provide similar basic features.

Description logics generally lie in the two-variable fragment of first-order logic with counting,[4] but have a special, yet intuitive, syntax. A description logic allows to model *individual elements*, which represent concrete objects, such as `bob` and `window1`; *concepts*, representing classes of individuals, such as `User` and `Component`; and *roles*, representing (binary) relations between individuals, such as `FocusesOn`. These semantic entities are syntactically described in axioms using *individual names*, *concept names*, and *role names*—which, respectively, correspond to constants, unary, and binary predicates in first-order logic.[5] Moreover, complex concept expressions can be constructed using the Boolean operators complement ($\neg$), intersection ($\sqcap$), and union ($\sqcup$); and *role restrictions*. For instance, the expression $\exists$`FocusesOn`.`Component` describes the class of all elements that focus on some component. Operators for constructing role expressions are not so common. Nevertheless, the inverse role operator ($\cdot^-$) represents a characteristic feature of *DL-Lite*. For example, applications for which there exists some element that focuses on them can be captured as follows:

$$\texttt{Application} \sqcap \exists \texttt{FocusesOn}^- . \top$$

where the concept $\top$ describes the class of all elements. The different DLs are characterized by the syntactic means they provide: the operators for specifying concepts and roles, and the kinds of axioms they allow for.

DL theories are called *knowledge bases* (KBs) and separate the axioms into an *ontology* and an *ABox*. While the ontology contains general domain knowledge, the ABox contains *data* about concrete objects and thus represents a description of (an extract of) the real world. Observe that the ABox can be seen as an instantiation of the global schema described in the previous section (see Figure 1.1). That is, the DL abstracts from the implementation aspect where the data is actually stored and does not consider the sources to be different from the global view. DL axioms are expressions of two kinds:

[2] A propositional logic is *complete* if every Boolean function can be expressed in a term using propositions that represent the arguments of the function.

[3] The "2" reflects the update of OWL [OWL12].

[4] There are a few exceptions, but these are rarely used today, such as DLs that allow for specifying transitive closures.

[5] The terms "concept" and "role" are generally used for both the syntactic entities, as abbreviations for "concept expression" and "role expression", and for the semantic entities.

- *Assertions*, such as $\texttt{User}(\texttt{bob})$ ("Bob is a user") and $\texttt{LooksAt}(\texttt{bob}, \texttt{window1})$ ("Bob looks at an element named `window1`"), occur in ABoxes and describe facts about concrete objects.
- *Inclusions* occur in ontologies and express is-a relations between concepts or roles; for example:

$$\texttt{VideoPlayer} \sqsubseteq \texttt{Application} \sqcap \texttt{EnergyIntensive} \sqcap \neg\texttt{SystemCritical}$$
$$\texttt{LooksAt} \sqsubseteq \texttt{FocusesOn}$$

  In natural language: "every video player is an energy-intensive, not system-critical application", and "every element that looks at something focuses on it".

Reasoning over DL knowledge bases originally often concentrated on ontologies and certain *standard reasoning problems*, such as the question whether a concept inclusion holds in any interpretation. For example, $\mathcal{EL}$ has been applied in terminological reasoning tasks such as the latter for a long time. Only recently, the growing importance of data in practice has lead to an increased interest in *query answering*. The latter usually denotes the task of answering queries over KBs with the goal of retrieving ABox data, and conjunctive queries currently represent one of the most important query languages in this context; though, note that the standard reasoning problems can also be considered as queries. The *DL-Lite* logics have been tailored to conjunctive query answering. This is reflected in the fact that, for many of them, conjunctive queries w.r.t. a KB can be rewritten into first-order queries encoding both the original CQ and the ontological knowledge in the way rewriting is described in the previous section. This turned out to be very efficient since the first-order queries can be represented in SQL and then be evaluated over a standard database containing the ABox data [Cal+17]. For example, if the inclusion $\texttt{LooksAt} \sqsubseteq \texttt{FocusesOn}$ is taken into account, then the CQ $\text{Context}_{\text{Focus}}$ (1.1) is rewritten into the following disjunction of CQs:

$$\Big(\exists y_1 y_2.\texttt{Component}(x_1) \wedge \texttt{Component}(y_1) \wedge \texttt{Component}(y_2) \wedge \texttt{PartOf}(x_1, y_1) \wedge \texttt{PartOf}(y_2, y_1) \wedge \texttt{FocusesOn}(x_2, y_2)\Big) \vee$$
$$\Big(\exists y_1 y_2.\texttt{Component}(x_1) \wedge \texttt{Component}(y_1) \wedge \texttt{Component}(y_2) \wedge \texttt{PartOf}(x_1, y_1) \wedge \texttt{PartOf}(y_2, y_1) \wedge \texttt{LooksAt}(x_2, y_2)\Big).$$

If this query then, in turn, is rewritten in terms of the data sources, then the observations from sensors of type `eye` are also taken into account for answering the CQ $\text{Context}_{\text{Focus}}$. This rewriting hence extends the standard database rewriting (1.2).

In fact, in several lightweight logics, ontology-based query answering can be rewritten into existing formalisms—but not always into first-order logic. This makes them especially interesting for applications, since mature tools for answering the rewritings often exist already. The practical importance of the lightweight logics is also reflected by the fact that the OWL standard has been complemented by three so-called OWL 2 profiles [Mot+12], which are subsets of OWL 2. Two of them, OWL 2 EL and OWL 2 QL are based on extensions of $\mathcal{EL}$ and a *DL-Lite* logic, respectively.

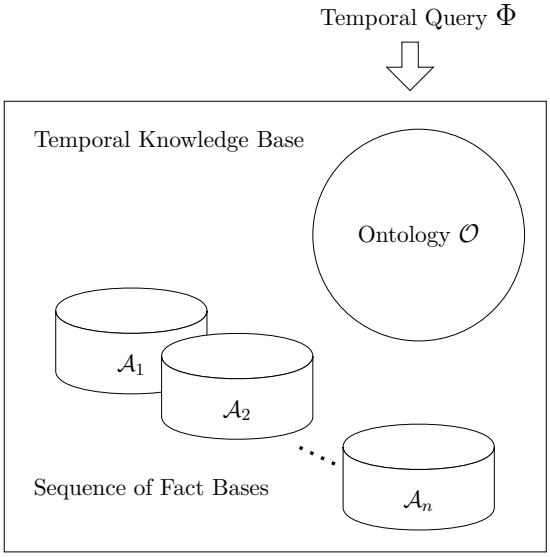

Figure 1.3: Our setting for ontology-based temporal query answering: a temporal query addressing a temporal knowledge base.

1.3 Ontology-Based Temporal Query Answering

The availability and importance of temporal data and ontologies in today's applications motivate our work on querying temporal data through classical ontologies. We investigate *temporal query languages*, which allow to refer to data associated to different moments in time, and regard *ontology-based temporal query answering* as reasoning problem. We focus on ontological axioms in *lightweight* logics, which allow for polynomial-time reasoning in the atemporal setting. Specifically, we regard the DLs $\mathcal{EL}$ and several *DL-Lite* fragments when studying complexity, but extend our results of the last chapter to various other logics.

Observe that temporal extensions of lightweight DLs where temporal operators may be applied to construct ontological concepts have turned out as being surprisingly complex, even undecidable [Art+07]. Nevertheless, research on such formalisms has been going on and identified "islands of tractability" and first-order rewritable formalisms, by restricting the available temporal operators and their applicability [Art+15a; GJK16]. The setting we consider is "easier" since, although we consider temporal data and queries, we do not allow the ontological axioms to contain temporal operators. In particular, decidability of this kind of ontology-based temporal query answering follows in most cases from results for more expressive formalisms [BGL12; BBL15b]. But it was open if the rather high complexities would decrease with lightweight logics. We provide results on the interaction of lightweight DLs and temporal logics and hence complement both strands of research.

The setting we focus on is depicted in Figure 1.3. The temporal data is represented through a *sequence of logical fact bases*, such as DL ABoxes, each of which contains facts about concrete objects and is associated to a specific point in time. General domain knowledge is described in an ontology and, in contrast to the facts, assumed

to hold at all time points. Together, the data and ontology form a *temporal knowledge base*. Note that we thus can represent a stream of data and, in line with this scenario, consider the queries to be answered over the whole sequence viewed from the *current time point n* ("now"). The *temporal queries* are formed by combining atemporal queries using Boolean operators and operators of *linear temporal logic* (LTL), such as $\Diamond_P$ ("at some time in the past") and $\Diamond_F$ ("at some time in the future"). Large parts of this work focus on *temporal conjunctive queries (TCQs)*, where the atemporal queries are CQs. For instance, a complex context where an energy-intensive application gets out of user focus, which might require a reconfiguration of the system (e.g., by decreasing quality parameters), can be encoded as a TCQ as follows, based on the CQ $\text{Context}_{\text{Focus}}$ (1.1):

$$\texttt{Application}(x_1) \land \texttt{Running}(x_1) \land \texttt{EnergyIntensive}(x_1) \land \neg\texttt{SystemCritical}(x_1) \land \Diamond_P \left(\text{Context}_{\text{Focus}}(x_1, x_2) \land \Diamond_F \left(\text{Context}_{\text{Focus}}(x_3, x_2) \land \neg\Diamond_F \text{Context}_{\text{Focus}}(x_1, x_2)\right)\right).$$

In natural language, the query describes a situation where, at some time in the past, the user x_2 focused on a component x_1, which is an energy-intensive application, running currently, and not system critical; and, at some time after that, the user has focused on a component x_3, and it is not the case that the user focus then or later was with x_1 again.

Observe that names such as `Running` and `FocusesOn` are used to describe dynamic knowledge, which may change over time. For describing knowledge that does not do so, certain names are often considered as *rigid* [Art+07; BBL15b]; for example, this would be adequate for the role name `PartOf`. We also consider rigid names. It may help to find additional inferences, but usually makes reasoning more complex.

In description logics, the investigation of ontology-based query answering in a temporal setting, targeting data retrieval, and focusing on decidable formalisms has started only in the recent past. Theoretical studies on complexity and rewritability have concentrated on qualitative temporal logics, such as LTL [GK12; Kla13; KM14b; Art+15a; BBL15b], and interval-based temporal logics [Art+14a; Art+15b]. Also our work has helped to advance the understanding of interactions between temporal queries and DL ontologies [BLT15; BT15b; BT15a]. This foundational work has recently lead to the consideration of metric temporal description logics in [GJO16; Baa+17], where the operators of LTL have been annotated with quantitative intervals, such as $\Diamond_F^{[0,5]}$ ("at some time within the next 5 time points"). This is an important feature to describe systems with discrete state changes, and hence data streams.[6]

The practical relevance of ontology-based query answering over temporal data has, in parallel, lead to significant implementation efforts. In the field of *stream reasoning* [CCG10; Ani+12; Kha+16; CMC16; Cal+17], the systems usually process a continuous stream of time-stamped RDF triples [CWL14] as $\langle(\texttt{bob}, \texttt{LooksAt}, \texttt{window1}), 11245\rangle$, for example, and often rely on DLs as ontology language (see [DDM16] for an introduction to the area and descriptions of further approaches). Although they offer expressive temporal query languages to extract finite sets of triples from the streams (e.g., languages with metric and aggregation operators), most systems then apply classical, atemporal techniques for reasoning and querying. Yet, especially recent approaches

[6] A detailed study of related work on temporal query answering in DLs is given at the end of Chapter 3, in Section 3.4.

also include the temporal dimension into reasoning [Ani+12; Kha+16]. The fact that, however, the "*systems are heterogeneous in terms of syntax, capabilities and evaluation semantics*" [Del+15, p. 353] has lead to a currently ongoing development of a unifying syntax and semantics [Bec+15; Del+15; Del+16].

Note that temporal extensions have also been proposed for the Semantic Web standards RDF, OWL, and SPARQL [The13] (e.g., in [GHV07; Mot12]) and there are several other areas where temporal query answering has been investigated regarding various kinds of ontologies. In database theory, temporal extensions of Datalog queries have been considered for a long time as so-called temporal deductive databases [BCW93] (see [AHV95] or Example 8.5 for a description of Datalog); the topic also has been taken up recently with a focus on rewritability [Kon+16]. Further, rule-based formalisms with temporal features have been studied regarding event processing [PK09]. However, many of these works allow the data to be augmented with temporal information of arbitrary kind and granularity, which yields a scenario only coarsely related to our setting. Furthermore, they do not focus on ontology-based data access through classical ontologies, but concentrate on the design of proprietary ontology languages which include temporal features.

Temporal query answering over standard databases represents another area of related work [CT05]. Next to query languages, data management and implementation aspects are in focus of the research there. Similar to our setting, where a finite sequence of fact bases models the data of the past and present, abstract temporal databases are represented as sequences of database instances over a given database schema. However, although there are some works that apply the open-world assumption, the closed-world assumption, considering the temporal databases to hold complete information about truth, is the quasi standard there [CT05]. Nevertheless, we can rely on results from that area when we consider algorithms and implementations of ontology-based temporal query answering (see Section 8.4). Note that, while there had been a gap between the theory and practice of temporal database systems for a long time [CT05], the importance of the temporal dimension is generally recognized today, and the possibility to associate data with temporal information has recently been incorporated into the SQL standard [KM12].

1.4 Contributions and Outline of the Thesis

The relevance and benefits of ontology-based data access have generally been recognized and are reflected in the increasing number of implementations. Due to the amounts of data to be processed and the efficiency requirements of applications, many of the systems focus on ontologies in lightweight logics today. While some of them—to a certain extent—even deal with temporal data already, research on the theoretical side is lacking behind. Temporal query languages that allow to access temporal data through classical ontologies in lightweight description logics have rarely been investigated yet.

The aim of this work is therefore to systematically analyze the interaction between LTL and lightweight description logic axioms to obtain (worst-case) complexity results for temporal query answering. To this end, we investigate the complexity of the corresponding decision problems, satisfiability and entailment. Furthermore, we want to de-

velop temporal query languages for which ontology-based query answering is rewritable into existing formalisms. The concrete research questions we focus on can be grouped into three areas:

LTL over lightweight description logic axioms What is the complexity of reasoning regarding temporal queries combining lightweight DL axioms via LTL operators? If necessary, can we find constraints for obtaining good results (i.e., matching those for LTL)? (Chapter 4) [BT15c; BT15b]

Entailment of temporal conjunctive queries What is the complexity of temporal conjunctive query entailment regarding ontologies in $\mathcal{EL}$, Horn fragments of *DL-Lite*, and more expressive *DL-Lite* logics? Are there logics for which we get tractable or rewritability results? (Chapters 5, 6, and 7) [BT15c; BT15b; BT15a]

Rewritability of temporal query answering How can we combine LTL operators with conjunctive queries to obtain rewritability results for temporal conjunctive query answering in lightweight DLs? Is it possible to extend the results to other temporal queries and logics? (Chapter 8) [BLT13a; BLT13b; BLT15; THÖ15]

Figures 4.1 and 9.1 present an overview of our complexity results. Our contributions are detailed in Section 3.3, and a comparative summary is given in Chapter 9. In particular, we defer the comparison of the results from the different chapters on temporal conjunctive query entailment to this chapter. Most of the results were obtained together with Stefan Borgwardt, and the results presented in Chapter 8 are joint work with Marcel Lippmann.

We close the introduction with some general guidelines for reading. We assume the reader to be familiar with *first-order logic* (FOL); in [Fit96], it is treated in detail. Basic knowledge in complexity theory is similarly presupposed. In particular, we consider finite state machines, Turing machines, and corresponding complexity classes without further explanation. Good introductions to the area also covering these topics are provided by [Pap94], the standard reference, and [AB09], a book containing also recent results.

Chapter 2 covers basic definitions and results regarding DLs, LTL, and computational complexity which we apply subsequently. These might be skipped by readers familiar with the topic. Chapter 3 contains a more specific introduction to temporal query answering. In particular, it introduces the query languages and problems investigated in Chapters 4 to 6, describes the technical challenges we solve, outlines some of our solutions, and gives an overview of related work. Apart from the dependencies on these preliminaries, the subsequent technical chapters, Chapters 4 to 8, are mostly self-contained. However, note that the methods applied in Chapters 4 to 7 are sometimes similar and, in the presented order, increase in intricacy. Chapter 9 summarizes the results, contains concluding remarks, and suggests directions of future work.

2 Basic Definitions

In this chapter, we provide basic definitions that are relevant throughout the work. The description logics we focus on and DL reasoning in the *classical*—in the sense of atemporal—setting are described in Section 2.1. In Section 2.2, we introduce linear temporal logic, the temporal component of the temporal queries we consider. In the last section, Section 2.3, we recapitulate basics of complexity theory.

2.1 Description Logics

The most important basics on description logics have already been introduced in Chapter 1. In this section, we provide formal definitions regarding the DL $\mathcal{EL}$ and several members of the *DL-Lite* family. In the following, we use the term *DL-Lite* for those members of the family this work focuses on; that is, the fragments *DL-Lite*$_{core}$, *DL-Lite*$^{\mathcal{H}}_{horn}$, *DL-Lite*$^{\mathcal{H}}_{krom}$, *DL-Lite*$^{\mathcal{H}}_{bool}$, and all variants in between, which are specified in this section. General syntax, semantics, and reasoning tasks are described first. Subsequently, we introduce reasoning with conjunctive queries. Then, we focus on the so-called *Horn* DLs among the DLs we consider, to review and prove certain properties they satisfy.

For a more detailed introduction to DLs, the interested reader is referred to the Description Logic Handbook [Baa+07], which covers the basics and includes advanced aspects of DL research.

2.1.1 Syntax, Semantics, and Standard Reasoning

As described in Chapter 1, description logics focus on *concepts*, which are interpreted as sets; *roles*, which are interpreted as binary relations; and *individual names*, which are interpreted as constants. Accordingly, DL *signatures* $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_R})$ are based on three kinds of non-logical symbols representing constants (i.e., zero-ary function symbols), unary, and binary predicates, respectively: *individual names* $\mathsf{N_I}$, *concept names* $\mathsf{N_C}$, and *role names* $\mathsf{N_R}$, all of which are non-empty, pairwise disjoint sets. The various DLs then differ in the allowed logical symbols and in the way the axioms of the theories are built.

In the following, we introduce the syntax of the axioms in the description logic $\mathcal{EL}$ and several members of the *DL-Lite* family and, based on the axioms, specify the notion of a DL theory, the *knowledge base*. In the remainder of the section, we cover the semantics and *standard reasoning tasks*.

Definition 2.1 (Syntax of Axioms in $\mathcal{EL}$) Let $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_R})$ be a DL signature.

In $\mathcal{EL}$, the sets of *roles* over Σ and *concepts* over Σ are defined, respectively, by the following grammars:

$$R ::= P \qquad\qquad C ::= \top \mid A \mid \exists R.D \mid D \sqcap E$$

where $A \in \mathsf{N_C}$, $P \in \mathsf{N_R}$, and D and E are concepts, in their turn; a concept of the form A, $\top$, $\exists R.\top$, or $\exists R.A$ is a *basic concept*.

In what follows, let $A_1, A_2, A_3 \in \mathsf{N_C} \cup \{\top\}$, $R \in \mathsf{N_R}$, B be a basic concept, and C and D be concepts. $\mathcal{EL}$ *axioms* are the following kinds of expressions: *concept inclusions (CIs)* of the form $C \sqsubseteq D$, and *assertions* of the form $C(a)$ and $R(a, b)$, where $a, b \in \mathsf{N_I}$. A CI is in *normal form* if it has one of the following forms:

$$A_1 \sqcap A_2 \sqsubseteq A_3, \ A_1 \sqsubseteq \exists R.A_2, \ B \sqsubseteq A_1. \qquad \diamondsuit$$

We sometimes use the abbreviation $\exists R_1 \dots R_\ell.C$ for the concept $\exists R_1. \dots \exists R_\ell.C$. Note that concept inclusions in $\mathcal{EL}$ or more expressive DLs are also called *general concept inclusions* (GCI), which expresses the fact that the inclusion may contain arbitrary concept expressions on the left-hand side—historically, first so-called primitive concept definitions with only concept names on the left-hand side were considered [Baa+07].

The logics of the *DL-Lite* family all extend the base formalism *DL-Lite*$_{core}$, in which CIs with complex concept expressions on the left-hand side cannot be expressed. In this work, we focus on several of the logics presented in [Art+09], which differ in the kind of concept inclusions, the Boolean operators allowed in the concept expressions, and if *role inclusion* axioms (also *role hierarchies*) are allowed. Similar to concept inclusion axioms, the latter are of the form $S \sqsubseteq R$ and express that the role S is more specific than the role R. *DL-Lite* fragments that allow for such inclusions are labeled with the superscript $\mathcal{H}$.

Definition 2.2 (Syntax of Axioms in *DL-Lite*) Let $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_R})$ be a DL signature. In *DL-Lite*, the sets of *roles* over Σ and *basic concepts* (also *concepts*) over Σ are defined, respectively, by the following grammars:[1]

$$R ::= P \mid P^- \qquad\qquad B ::= A \mid \exists R.\top$$

where $A \in \mathsf{N_C}$, $P \in \mathsf{N_R}$, and $\cdot^-$ denotes the inverse role operator. $\mathsf{N_R^-}$ denotes the set of roles.

DL-Lite axioms are the following kinds of expressions: *concept inclusions (CIs)* of the form

$$B_1 \sqcap \dots \sqcap B_m \sqsubseteq B_{m+1} \sqcup \dots \sqcup B_{m+n} \qquad (*)$$

where $B_1, \dots, B_{m+n}$ are concepts;[2] *role inclusions (RIs)* of the form $S \sqsubseteq R$, where $R, S \in \mathsf{N_R^-}$; and *assertions* of the form $B(a)$ and $P(a, b)$, where B is a basic concept, $P \in \mathsf{N_R}$, and $a, b \in \mathsf{N_I}$.

For $c \in \{core, horn, krom, bool\}$, we denote by *DL-Lite*$_c$ the logic that does not allow for role inclusions and restricts concept inclusions of the form $(*)$ as follows:

- m, n are arbitrary if $c = bool$,
- $m + n \leq 2$ if $c = krom$,
- $n \leq 1$ if $c = horn$,

[1] In the literature, a *role* is sometimes an expression that may be prefixed by negation [Cal+07b].
[2] Both sides of CIs may be empty.

- $m + n \leq 2$ and $n \leq 1$ if $c = \mathit{core}$.

If role inclusions are allowed in addition, this is indicated by the superscript $\mathcal{H}$, and we obtain the four DLs denoted by $\mathit{DL\text{-}Lite}_c^{\mathcal{H}}$. $\diamondsuit$

In *DL-Lite*, the abbreviation $\exists R$ is usually used to abbreviate concepts of the form $\exists R.\top$, where R is a role. As usual, we generally denote the empty conjunction ($\sqcap$) by $\top$ and the empty disjunction ($\sqcup$) by $\bot$. We may further use the abbreviations $B_1 \sqcap \cdots \sqcap B_m \sqsubseteq \neg B$ for $B_1 \sqcap \cdots \sqcap B_m \sqcap B \sqsubseteq \bot$, $\bigsqcap \mathcal{B}$ for the conjunction $B_1 \sqcap \cdots \sqcap B_m$ if $\mathcal{B} = \{B_1, \ldots, B_m\}$, $P^-(a, b) := P(b, a)$, and $(P^-)^- := P$ for $P \in \mathsf{N_R}$ and $a, b \in \mathsf{N_I}$.

In some constructions (in Chapters 4 and 6), we also consider *negated* assertions of the form $\neg\alpha$, where α is an assertion; if this is the case, it is mentioned explicitly.

Definition 2.3 (Syntax of Knowledge Bases) Let $\mathcal{DL}$ be a description logic. An *ontology* written in $\mathcal{DL}$ is a finite set of concept and (if allowed in $\mathcal{DL}$) role inclusions, and an *ABox* is a finite set of assertions of concept and role names. Together, an ontology $\mathcal{O}$ and an ABox $\mathcal{A}$ form a *knowledge base* (KB) $\mathcal{K} := \mathcal{O} \cup \mathcal{A}$, written $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$. $\diamondsuit$

We sometimes also refer to the ABox as *fact base* or simply as the *data*.[3] In this work, we assume every knowledge base to be such that all concept and role names occurring in the ABox also occur in the ontology. Given the (standard) semantics introduced below, it can readily be checked that this assumption is without loss of generality

For a given KB $\mathcal{K} := \mathcal{O} \cup \mathcal{A}$, we denote by $\mathsf{N_I}(\mathcal{K})$ and $\mathsf{N_I}(\mathcal{A})$ the set of individual names that occur in $\mathcal{K}$ and $\mathcal{A}$, respectively (i.e., in $\mathcal{EL}$ and *DL-Lite*, we have $\mathsf{N_I}(\mathcal{K}) = \mathsf{N_I}(\mathcal{A})$); by $\mathsf{N_C}(\mathcal{O})$ and $\mathsf{N_R}(\mathcal{O})$ the sets of, respectively, concept names and role names occurring in $\mathcal{K}$; by $\mathsf{N_R^-}(\mathcal{O})$ the set of roles occurring in $\mathcal{K}$ if it is in *DL-Lite*; and by $\mathbb{S}(\mathcal{O})$ the set of all concepts that occur in $\mathcal{O}$. Note that the latter set includes all sub-concepts of complex concept expressions. A *concept over* $\mathcal{O}$ is a concept constructed (only) from the concept and role names occurring in $\mathcal{O}$; observe that it does not necessarily have to be contained in $\mathbb{S}(\mathcal{O})$. Moreover, $\mathbb{B}(\mathcal{O})$ denotes the set of all basic concepts that can be built from $\mathsf{N_C}(\mathcal{O})$, $\top$, and the roles occurring in $\mathcal{O}$, and $\mathbb{B}^\neg(\mathcal{O})$ denotes the set $\mathbb{B}(\mathcal{O})$ extended by negation, meaning $\mathbb{B}^\neg(\mathcal{O}) := \{B, \neg B \mid B \in \mathbb{B}(\mathcal{O})\}$. Note that, regarding *DL-Lite*, $\mathbb{S}(\mathcal{O})$ and $\mathbb{B}(\mathcal{O})$ nearly coincide; yet, $\mathbb{B}(\mathcal{O})$ *always* contains $\top$ and the concept $\exists P^-$ for all $P \in \mathsf{N_R}(\mathcal{O})$.

The semantics of DLs is commonly specified in a model-theoretic way, based on interpretations. General logical notions like consistency and entailment can hence be defined as usual.

Definition 2.4 (Semantics) An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ for a description logic signature $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_R})$ consists of a non-empty set $\Delta^\mathcal{I}$, the *domain* of $\mathcal{I}$, and an *interpretation function* $\cdot^\mathcal{I}$, which assigns to every $A \in \mathsf{N_C}$ a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, to every $P \in \mathsf{N_R}$ a binary relation $P^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, and to every $a \in \mathsf{N_I}$ an element $a^\mathcal{I} \in \Delta^\mathcal{I}$ such that, for all $a, b \in \mathsf{N_I}$ with $a \neq b$, we have $a^\mathcal{I} \neq b^\mathcal{I}$ (*unique name assumption* (UNA)). This function is extended to all roles and concepts as described in the first part of Figure 2.1.

[3] In correspondence with the notion of ABox, ontologies are often separated into *TBoxes* and *RBoxes*, containing the concept and role inclusions, respectively.

Name	Syntax	Semantics
inverse role	R^-	$\{(y,x) \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid (x,y) \in R^\mathcal{I}\}$
top concept	$\top$	$\Delta^\mathcal{I}$
bottom concept	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^\mathcal{I} \setminus C^\mathcal{I}$
conjunction	$C \sqcap D$	$C^\mathcal{I} \cap D^\mathcal{I}$
disjunction	$C \sqcup D$	$C^\mathcal{I} \cup D^\mathcal{I}$
exist. restriction	$\exists R.C$	$\{x \in \Delta^\mathcal{I} \mid \exists y \in C^\mathcal{I}, (x,y) \in R^\mathcal{I}\}$
concept inclusion	$C \sqsubseteq D$	$C^\mathcal{I} \subseteq D^\mathcal{I}$
role inclusion	$R \sqsubseteq S$	$R^\mathcal{I} \subseteq S^\mathcal{I}$
concept assertion	$B(a)$	$a^\mathcal{I} \in B^\mathcal{I}$
role assertion	$R(a,b)$	$(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$

Figure 2.1: Semantics of role expressions, concept expressions, and axioms for an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$.

An interpretation $\mathcal{I}$ *satisfies* (or is a *model* of) an axiom α, written $\mathcal{I} \models \alpha$, if the corresponding condition given in Figure 2.1 is satisfied. $\mathcal{I}$ *satisfies* (or is a *model* of) a knowledge base $\mathcal{K}$, written $\mathcal{I} \models \mathcal{K}$, if it satisfies all axioms contained in it.

A knowledge base $\mathcal{K}$ is *consistent* (or *satisfiable*) if it has a model, and it is *inconsistent* (or *unsatisfiable*) otherwise. $\mathcal{K}$ *entails* an axiom α, written $\mathcal{K} \models \alpha$, if all models of $\mathcal{K}$ also satisfy α. This terminology and notation is extended to (single) axioms, ontologies, and ABoxes by regarding each as a (singleton) knowledge base. ♢

We denote the fact that an interpretation $\mathcal{I}$ does not satisfy a KB $\mathcal{K}$ by $\mathcal{I} \not\models \mathcal{K}$ and, similarly, non-entailment by $\mathcal{K} \not\models \alpha$. In accordance with Figure 2.1, the negated assertions of the form $\neg B(a)$ and $\neg R(a,b)$, which we sometimes consider, are satisfied in an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ if, respectively, $a^\mathcal{I} \notin B^\mathcal{I}$ and $(a^\mathcal{I}, b^\mathcal{I}) \notin R^\mathcal{I}$ hold.

Regarding two domain elements d and e and an interpretation $\mathcal{I}$ such that $(d,e) \in R^\mathcal{I}$, d is an R-predecessor of e, and e an R-successor of d. Note that the terms "individual elements", "domain elements", "elements", and "individuals" are used interchangeably for the elements of an interpretation domain. If the terms are prefixed by "named", then we refer to those elements of the domain that are used to interpret individual names.[4] In what follows, the signature of an interpretation is generally not mentioned explicitly if it is irrelevant or clear from the context.

In some constructions, we apply the DL $\mathcal{ELO}_\bot$, which extends $\mathcal{EL}$ by allowing $\bot$ and so-called *nominals* in concept expressions. $\bot$ is interpreted as the empty set and can be used in an ontology for expressing disjointness of concepts. Nominals are concepts of the form $\{a\}$, based on some individual name a, and interpreted as singleton sets that contain the corresponding named individual. Further, note that *DL-Lite*$_R$, which is the DL closest to the OWL 2 QL profile, extends *DL-Lite*$^\mathcal{H}_{core}$ in that it allows to express disjointness of roles.

[4] In the literature, the term "individuals" sometimes only refers to those elements of the domain that are used to interpret individual names.

Given the semantics, observe that allowing conjunction on the right-hand side of CIs does not increase expressivity since any CI of the form $C \sqsubseteq D \sqcap E$ can be split into two CIs $C \sqsubseteq D$ and $C \sqsubseteq E$ in a KB without affecting the semantics. For similar reasons, we can assume all CIs to have maximally two conjuncts on the left-hand side and maximally two disjuncts on the right-hand side.

Ontologies and knowledge bases on the whole are usually not only used for modeling domain knowledge and querying it, but also for deriving logical consequences that are not explicitly stated in the stored knowledge. This (typically automatic) process is called *reasoning* and comprises several *standard reasoning problems*. Below, we list those relevant for our work; for a larger overview, we refer to [Baa+07].

Definition 2.5 (Standard Reasoning Problems) Let $\mathcal{K}$ be a DL knowledge base, and let C and D be concepts. The *standard reasoning problems* in DLs include the following:

- *Concept Subsumption:* Does $\mathcal{K} \models C \sqsubseteq D$ hold?
- *Concept Satisfiability:* Is there an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$ and $C^{\mathcal{I}} \neq \emptyset$?
- *Consistency Checking:* Is $\mathcal{K}$ consistent?
- *Instance Checking:* Does $\mathcal{K} \models C(a)$ hold? ♢

It is well-known that these reasoning tasks are reducible to each other in linear time in any DL that allows for concept name assertions and CIs expressing disjointness, of the form $C \sqcap D \sqsubseteq \bot$ (e.g., a concept C is not satisfiable w.r.t. a KB $\mathcal{K}$ if $\mathcal{K} \models C \sqsubseteq \bot$; that CI is equivalent to $C \sqcap \top \sqsubseteq \bot$).

In contrast, *conjunctive query answering* and *entailment* are non-standard reasoning problems. These problems are important for TCQ answering since TCQs are based on conjunctive queries. We therefore introduce them next.

2.1.2 Conjunctive Queries

Conjunctive query answering is a core functionality of database systems and increasingly studied in DLs recently. The goal is to provide ontology-based query answering on top of databases. We in the following define conjunctive queries and the more general unions of conjunctive queries, and introduce the reasoning problems relevant in this work.

Definition 2.6 (Syntax of Unions of Conjunctive Queries) Let $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_R})$ be a DL signature, $\mathsf{N_V}$ be the set of variables, and $\mathsf{N_T} := \mathsf{N_I} \cup \mathsf{N_V}$ be the set of terms.

A *conjunctive query* (CQ) over Σ is of the form $\varphi = \exists y_1, \ldots, y_m.\psi$, where $y_1, \ldots, y_m \in \mathsf{N_V}$ and ψ is a (possibly empty) finite conjunction of atoms as follows:

- $A(t)$ (*concept atom*) with $A \in \mathsf{N_C}$ and $t \in \mathsf{N_T}$,
- $R(s,t)$ (*role atom*) with $R \in \mathsf{N_R}$ and $s, t \in \mathsf{N_T}$.

The variables occurring in φ that are different from $y_1, \ldots, y_m$ are *free variables*.

A *union of conjunctive queries* (UCQ) is a disjunction of CQs. The *free variables* of a UCQ are the union of all free variables of its disjuncts. ♢

We denote the set of individuals occurring in a UCQ φ by $\mathsf{N_I}(\varphi)$, the set of variables occurring in φ by $\mathsf{N_V}(\varphi)$, and the set of terms occurring in φ by $\mathsf{N_T}(\varphi)$. $\mathsf{At}(\varphi)$ denotes the set of atoms occurring in φ. In the context of *DL-Lite*, we may use an expression of the form $R^-(s,t)$ to denote the role atom $R(t,s)$, as with role assertions; further, note that the definition of CQs does not allow basic concepts of the form $\exists R(x), R \in \mathsf{N_R^-}$, to occur in CQs. We sometimes though regard such CQs, which is possible since such atoms can obviously be replaced by atoms of the form $R(x,y)$ if $R \in \mathsf{N_R}$ and $R(y,x)$ otherwise, if the set of existentially quantified variables of φ is extended with a fresh variable y, correspondingly. We may write $\bigwedge \mathcal{B}(x)$ for the conjunction $B_1(x) \wedge \cdots \wedge B_\ell(x)$ if $\mathcal{B} = \{B_1, \ldots, B_\ell\}$.

Since we focus on *Boolean* queries (i.e., queries without free variables), we define the semantics only for those. As usual, this is done in a model-theoretic way, based on the notion of homomorphisms [CM77].

Definition 2.7 (Semantics of Unions of Conjunctive Queries) Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation and φ be a Boolean CQ. A mapping $\pi \colon \mathsf{N_T}(\varphi) \to \Delta^\mathcal{I}$ is a homomorphism of φ into $\mathcal{I}$ if

- $\pi(a) = a^\mathcal{I}$ for all $a \in \mathsf{N_I}(\varphi)$,
- $\pi(t) \in A^\mathcal{I}$ for all concept atoms $A(t)$ in φ, and
- $(\pi(s), \pi(t)) \in R^\mathcal{I}$ for all role atoms $R(s,t)$ in φ.

$\mathcal{I}$ *satisfies* (or is a *model* of) φ, written $\mathcal{I} \models \varphi$, if there is such a homomorphism. $\mathcal{I}$ *satisfies* (or is a *model* of) a UCQ ψ, written $\mathcal{I} \models \psi$, if it satisfies one of its disjuncts.

ψ is *entailed* by a KB $\mathcal{K}$, written $\mathcal{K} \models \psi$, if every model of $\mathcal{K}$ is also a model of ψ. $\diamondsuit$

Regarding UCQs, DL research primarily focuses on the following reasoning problems.

Definition 2.8 (Reasoning Problems for UCQs) For a given Boolean UCQ φ, UCQ ψ, and KB $\mathcal{K}$, there are the following reasoning problems:

- *UCQ Entailment:* Does $\mathcal{K} \models \varphi$ hold?
- *UCQ Answering*: Determine all assignments $\mathfrak{a}$ (called *certain answers*) of the free variables in the query to named individuals occurring in $\mathcal{K}$ such that $\mathcal{K} \models \mathfrak{a}(\psi)$. $\diamondsuit$

Our focus on Boolean CQs (and UCQs) affects the query answering problem in that we consider a single such assignment at most; that is, the set of all answers is $\{()\}$ if $\mathcal{K} \models \varphi$ and $\emptyset$ otherwise.

To ease presentation, we sometimes refer to a CQ as a set, thereby meaning the set of all of its atoms, or as a graph.

Definition 2.9 (Graph of a CQ) The *graph* G_φ of a CQ φ is the graph $\mathrm{G}_\varphi = (V, E)$ where the set of nodes is defined as $V := \mathsf{N_T}(\varphi)$, the edges are given by the role atoms such that $E := \{(s,t) \mid R(s,t) \in \varphi\}$, and a labeling function ℓ_φ maps every node (edge) to a set of concepts (roles) such that $\ell_\varphi(s) := \{A \mid A(s) \in \varphi\}$ $(\ell_\varphi((s,t)) := \{R \mid R(s,t) \in \varphi\})$.

A CQ φ is *connected* if G_φ is connected. Terms s and t are *connected* in a CQ if it contains role atoms $R_1(s, u_1), R_2(u_2, u_3), \ldots, R_\ell(u_\ell, t)$, where $R_1, \ldots, R_\ell$ are roles and $u_1, \ldots, u_\ell \in \mathsf{N_T}$, and *directly* connected if they occur together in a role atom.

A set of CQs $\varphi_1, \ldots, \varphi_\ell$ is a *partition* of φ if the graphs $\mathrm{G}_{\varphi_1}, \ldots, \mathrm{G}_{\varphi_\ell}$ represent a partition of G_φ and, for all $i \in [1, \ell]$ and elements x in the domain of ℓ_{φ_i}, $\ell_\varphi(x) = \ell_{\varphi_i}(x)$.$\diamondsuit$

Note that the latter condition is necessary because the graph representation does not capture the concept atoms.

In general, the task of checking conjunctive query entailment is rather complex. For several description logics, the so-called *Horn* DLs, there are however practical algorithms. These are often based on *canonical interpretations*.

2.1.3 Canonical Interpretations for Horn Description Logics

In this section, we focus on the *Horn* DLs we introduced in Section 2.1.1, $\mathcal{EL}$ and *DL-Lite*$^{\mathcal{H}}_{horn}$. In a nutshell, Horn DLs do not allow to express disjunction on the right-hand side of CIs (i.e., neither directly in the syntax, nor indirectly through the semantics), such that the CIs can be represented as first-order Horn clauses. Reasoning in these DLs is easier than in general since it can be done using deterministic algorithms, which are often based on *canonical interpretations*. In what follows, we recall the well-known construction of these interpretations for knowledge bases in $\mathcal{EL}$ and *DL-Lite*$^{\mathcal{H}}_{horn}$ together with important properties of them and prove additional such properties.

In a nutshell, the canonical interpretation of a knowledge base $\mathcal{K}$ captures exactly the information described by $\mathcal{K}$. Since this is the knowledge to be satisfied in every model of $\mathcal{K}$, the canonical interpretation can be used for checking the consistency of $\mathcal{K}$ and for answering CQs w.r.t. $\mathcal{K}$—by checking whether the canonical interpretation is a model of $\mathcal{K}$ and if it satisfies a CQ, respectively. We provide different definitions depending on the logic to facilitate later proofs. For $\mathcal{EL}$, we directly consider the knowledge entailed from $\mathcal{K}$, as it is done in [LTW09; KRH07], for example. Regarding *DL-Lite*, we explicitly construct the interpretation similar to [Cal+07b; BAC10], by applying the standard *chase* procedure for obtaining models of knowledge bases [DNR08]. We use this definition in order to be able to refer to the results of [BAC10], which hold for the non-standard setting with negative assertions in ABoxes, which we focus on. Note that [BAC10] extend the original definition of [Cal+07b] regarding *DL-Lite*$^{\mathcal{H}}$ to the logic *DL-Lite*$^{\mathcal{H}}_{horn}$, and we further extend it. In particular, our canonical interpretation contains (unnamed) prototypical R-successors with $R \in \mathsf{N_R^-}$ for all elements required to satisfy $\exists R$, by the knowledge base. In contrast, [Cal+07b; BAC10] only consider such prototypical successors if the knowledge base (i.e., the corresponding ABox) does not already identify a named individual to be such a successor. Unlike us, [Cal+07b; BAC10] neither consider arbitrary basic concept assertions, but only concept names.

The prototypical domain elements are of the form u_ϱ (u for "unnamed"), where ϱ is a *path* $\varrho := aR_1C_1 \ldots R_\ell C_\ell$ over symbols occurring in the knowledge base with a being an individual name, $R_1, \ldots, R_\ell$ being roles, and $C_1, \ldots, C_\ell$ being concepts from $\mathbb{S}(\mathcal{O})$. We assume that the knowledge base does not already contain symbols of the form of these elements. $|\varrho| := \ell$ denotes the *length* of a path as ϱ. Observe that such a path also specifies the interpretation of the symbols contained in it in that it describes the exis-

tence of domain elements $a, u_{aR_1C_1}, \ldots, u_{aR_1C_1 \ldots R_\ell C_\ell}$, which are related via $R_1, \ldots, R_\ell$, respectively; and each element $u_{aR_1C_1 \ldots R_mC_m}$ for $m \leq \ell$ satisfies C_m. Furthermore, u_ϱ is only contained in the domain of the canonical interpretation if relations as described by ϱ have to be present in every model of the knowledge base. Note that *DL-Lite* ontologies can only enforce unnamed elements of the form $u_{aR_1\top \ldots R_\ell\top}$ to exist, because they do not allow for qualified existential restriction on the right-hand side of CIs. We therefore usually write $u_{aR_1 \ldots R_\ell}$ in that context, to simplify notation.

Definition 2.10 (Canonical Interpretation in $\mathcal{EL}$) Let $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ be an $\mathcal{EL}$ knowledge base. We first define the set

$$\Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}} := \bigcup_{j=0}^{\infty} \Delta_{\mathsf{u}}^{j}$$

where

$$\begin{aligned} \Delta_{\mathsf{u}}^{0} &:= \{u_{aRC} \mid a \in \mathsf{N_I}(\mathcal{K}),\ C \in \mathbb{S}(\mathcal{O}),\ \mathcal{K} \models \exists R.C(a)\}, \\ \Delta_{\mathsf{u}}^{j+1} &:= \{u_{\varrho R_1C_1R_2C_2} \mid \exists u_{\varrho R_1C_1} \in \Delta_{\mathsf{u}}^{j}, \mathcal{O} \models C_1 \sqsubseteq \exists R_2.C_2\}. \end{aligned}$$

The canonical interpretation $\mathcal{I}_\mathcal{K}$ for $\mathcal{K}$ is defined as follows, for all $a \in \mathsf{N_I}(\mathcal{A})$, $A \in \mathsf{N_C}$, and $R \in \mathsf{N_R}$:

$$\begin{aligned} \Delta^{\mathcal{I}_\mathcal{K}} &:= \mathsf{N_I}(\mathcal{A}) \cup \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}}, \\ a^{\mathcal{I}_\mathcal{K}} &:= a, \\ A^{\mathcal{I}_\mathcal{K}} &:= \{a \in \mathsf{N_I}(\mathcal{A}) \mid \mathcal{K} \models A(a)\} \cup \{u_{\varrho RC} \in \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}} \mid \mathcal{O} \models C \sqsubseteq A\}, \\ R^{\mathcal{I}_\mathcal{K}} &:= \{(a,b) \mid R(a,b) \in \mathcal{A}\} \cup \{(a, u_{aRC}) \in \mathsf{N_I}(\mathcal{A}) \times \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}}\} \cup \\ &\qquad \{(u_\varrho, u_{\varrho RC}) \in \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}} \times \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}}\}. \end{aligned}$$

The *finite* canonical interpretation $\mathcal{I}_\mathcal{K}^f$ is defined correspondingly, but based on elements of the form u_C for all $C \in \mathbb{S}(\mathcal{O})$ instead of on different elements of the form $u_{\varrho C}$. Analogously, these elements are collected in the set $\Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}^f}$. ♢

For simplicity, we assume in the definition regarding *DL-Lite* that, if the RI $S \sqsubseteq R$ is contained in an ontology $\mathcal{O}$, then we also have $\exists S \sqsubseteq \exists R \in \mathcal{O}$ and $\exists S^- \sqsubseteq \exists R^- \in \mathcal{O}$; and that $\mathcal{O}$ contains all trivial axioms of the form $B \sqsubseteq B$ for $B \in \mathbb{B}(\mathcal{O})$.

Definition 2.11 (Canonical Interpretation in $\mathcal{DL\text{-}Lite}_{horn}^{\mathcal{H}}$) Let $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ be a $\mathcal{DL\text{-}Lite}_{horn}^{\mathcal{H}}$ knowledge base. First, for all $A \in \mathsf{N_C}$ and $P \in \mathsf{N_R}$, we define:

$$\begin{aligned} A^0 &:= \{a \mid A(a) \in \mathcal{A}\}, \\ P^0 &:= \{(a,b) \mid P(a,b) \in \mathcal{A}\} \cup \\ &\qquad \{(a, u_{aP}) \mid \exists P(a) \in \mathcal{A}\} \cup \{(u_{aP^-}, a) \mid \exists P^-(a) \in \mathcal{A}\}. \end{aligned}$$

Then, iterate over all $i \geq 0$: for all $X \in \mathsf{N_C} \cup \mathsf{N_R}$ define $X^{i+1} := X^i$; apply one of the following rules for all $A \in \mathsf{N_C}$, $R, S \in \mathsf{N_R^-}$, and $B, B_1, B_2 \in \mathbb{B}(\mathcal{O})$; and increment i; $(d,e) \in (P^-)^i$ for $P \in \mathsf{N_R}$ denotes the fact that $(e,d) \in P^i$, and $d \in (\exists R)^i$ denotes the existence of an element e such that $(d,e) \in R^i$:

- If $B_1 \sqcap B_2 \sqsubseteq A \in \mathcal{O}$, $A \in \mathsf{N_C}$, and $e \in B_1^i \cap B_2^i$, then add e to A^{i+1}.
- If $B \sqsubseteq \exists R \in \mathcal{O}$ and $e \in B^i$:
  - if $e \in \mathsf{N_I}(\mathcal{A})$, then add (e, u_{eR}) to R^{i+1};
  - if $e = u_\varrho$, then add $(e, u_{\varrho R})$ to R^{i+1}.
- If $\exists R \sqsubseteq A \in \mathcal{O}$, $(d, e) \in R^i$, then add d to A^{i+1}.
- If $S \sqsubseteq R \in \mathcal{O}$ and $(d, e) \in S^i$, then add (d, e) to R^{i+1}.

The set $\Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}}$ collects the above introduced unnamed individuals.

The *canonical interpretation* $\mathcal{I}_\mathcal{K}$ for $\mathcal{K}$ is then defined as follows, for all $a \in \mathsf{N_I}(\mathcal{A})$, $A \in \mathsf{N_C}$, and $P \in \mathsf{N_R}$:

$$\Delta^{\mathcal{I}_\mathcal{K}} := \mathsf{N_I}(\mathcal{A}) \cup \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}}, \qquad a^{\mathcal{I}_\mathcal{K}} := a,$$
$$A^{\mathcal{I}_\mathcal{K}} := \bigcup_{i=0}^{\infty} A^i, \qquad P^{\mathcal{I}_\mathcal{K}} := \bigcup_{i=0}^{\infty} P^i. \qquad \diamondsuit$$

Note that the above assumptions about additional axioms in the ontology ensure that, whenever there is a named element $a \in (\exists R)^i$ for some $i \geq 0$, then a has an R-successor of the form u_{aR} in the canonical interpretation, and similar for the unnamed elements.

In the remainder of this section, we recall and prove results about how the canonical interpretation may simplify reasoning. To this end, we assume $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ to be a consistent knowledge base in $\mathcal{EL}$ or *DL-Lite*, depending on the context. In the latter case, $\mathcal{A}$ may particularly include negated assertions.

Regarding $\mathcal{EL}$, we refer to the results from [LTW09], where the canonical interpretation is however defined slightly differently, based on the finite one. More precisely, the possible paths over the domain elements of the latter that start at the named individuals represent the domain elements of the former interpretation. It is easy to see that this approach yields an interpretation corresponding to the one from Definition 2.10. Regarding *DL-Lite*, note that the two above mentioned differences, w.r.t. basic concept assertions and the additional successor individuals we consider, do not have special effects on reasoning. This is why we below refer to the results of [BAC10] without providing detailed proofs.

An important property of the canonical interpretation $\mathcal{I}_\mathcal{K}$, which can be checked easily, is that it is a model of $\mathcal{K}$. If $\mathcal{K}$ was inconsistent, then this would obviously not be the case. The converse of this statement is however harder to show—for *DL-Lite*; $\mathcal{EL}$ KBs cannot be inconsistent. The proof proposed by [Cal+07b; BAC10] is three-fold:

- First, it is shown that $\mathcal{I}_\mathcal{K}$ is a model of all *positive inclusions* in $\mathcal{O}$, which are CIs whose right-hand side is not $\bot$; all other CIs are called *negative inclusions*.
- For checking satisfiability of *DL-Lite*$_{horn}^{\mathcal{H}}$ KBs, negative inclusions are critical: if a negative inclusion in the ontology is violated by assertions of the ABox, then the knowledge base is inconsistent and hence unsatisfiable. In particular, an interaction of positive and negative inclusions may lead to such an inconsistency. This is why a special *closure* of the negative inclusions contained in $\mathcal{O}$ is regarded, which represents all negative inclusions implied by the ontology.

The second step then consists of showing that $\mathcal{K}$ is consistent iff the assertions of the ABox do not contradict this closure.

- Third and last, it is shown that the latter is the case iff $\mathcal{I}_\mathcal{K}$ is a model of $\mathcal{K}$.

These observations show that neither basic concept assertions nor "unnecessary" prototypical successors have special effects on the outcomes of the proof in [BAC10]. We hence can similarly state the result.

Lemma 2.12 ([BAC10, Lem. 3, Thm. 4],[LTW09, Prop. 1]) *$\mathcal{I}_\mathcal{K} \models \mathcal{K}$ and $\mathcal{I}_\mathcal{K}^f \models \mathcal{K}$.*

A result equally important for our work is about queries.

Lemma 2.13 ([BAC10, Thm. 9],[LTW09, Prop. 4]) *For every Boolean UCQ φ, we have $\mathcal{K} \models \varphi$ iff $\mathcal{I}_\mathcal{K} \models \varphi$.* □

In the following Lemmas 2.14 and 2.15, we describe the concepts satisfied by the domain elements of the canonical model and the subset of prototypical elements, respectively. Note that the following Lemma 2.14 is restricted to *DL-Lite*.

Lemma 2.14 *Regarding Definition 2.11, all $e \in \Delta^{\mathcal{I}_\mathcal{K}}$, and all $B \in \mathbb{B}(\mathcal{O})$, we have $e \in B^{\mathcal{I}_\mathcal{K}}$ iff $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq B$, where $\mathcal{B}$ is defined as follows, based on the minimal number i for which there is a symbol X such that $e \in X^i$:*

$$\mathcal{B} := \{A \in \mathsf{N_C}(\mathcal{O}) \mid e \in A^i\} \cup \{\exists R \mid R \in \mathsf{N_R^-}(\mathcal{O}),\ (e,d) \in R^i\}.$$

Proof. ($\Leftarrow$) For all $C \in \mathcal{B}$, we know that $e \in C^{\mathcal{I}_\mathcal{K}}$, by the definitions of $\mathcal{B}$ and $\mathcal{I}_\mathcal{K}$. Hence, Lemma 2.12 yields the claim.

($\Rightarrow$) Let j be the minimal index for which $e \in B^j$, which means that $j \geq i$. We show the claim by induction on j. If $j = i$, then $B \in \mathcal{B}$, and hence $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq B$ trivially holds.

Regarding $j > i$, we assume that the claim holds for all $C \in \mathbb{B}(\mathcal{O})$ for which $e \in C^{j-1}$. We consider the rule in Definition 2.11 that causes e to be contained in B^j.

- If it is because of a CI $\bigsqcap \mathcal{B}' \sqsubseteq B \in \mathcal{O}$, then we have $e \in C^{j-1}$ for all $C \in \mathcal{B}'$. Hence, the induction hypothesis together with the semantics yields $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \bigsqcap \mathcal{B}'$. Because of the considered CI, this leads to $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq B$.

  For the other kinds of CIs, the proof works correspondingly.

- If it is because of the last rule, some $S \sqsubseteq R \in \mathcal{O}$, and $(e,d) \in S^{j-1}$ ($(d,e) \in S^{j-1}$), then B must be of the form $B = \exists R^{(-)}$. Further, $(e,d) \in S^{j-1}$ ($(d,e) \in S^{j-1}$) implies $e \in (\exists S^{(-)})^{j-1}$, for which the induction hypotheses yields $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S^{(-)}$. By our assumption (see the part above Definition 2.11), we have $\exists S \sqsubseteq \exists R \in \mathcal{O}$ ($\exists S^- \sqsubseteq \exists R^- \in \mathcal{O}$) and thus get $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R^{(-)}$. □

The next lemma describes the concepts the new domain elements in $\Delta_\mathsf{u}^{\mathcal{I}_\mathcal{K}}$ satisfy in a straightforward way and hence shows that an element of the form $u_{\varrho RC} \in \Delta_\mathsf{u}^{\mathcal{I}_\mathcal{K}}$ can indeed serve as a prototypical R-successor. Regarding *DL-Lite*, the lemma directly follows from Definition 2.11 and Lemma 2.14. For $\mathcal{EL}$, it is easy to prove by induction on the structure of concepts.

Lemma 2.15 *For all $u_{\varrho RC} \in \Delta_{\mathsf{u}}^{\mathcal{I}_\mathcal{K}}$ and $D \in \mathbb{S}(\mathcal{O})$, we have $u_{\varrho RC} \in D^{\mathcal{I}_\mathcal{K}}$ iff*

- *$\mathcal{O} \models C \sqsubseteq D$, if $\mathcal{K}$ is in $\mathcal{EL}$, and*
- *$\mathcal{O} \models \exists R^- \sqsubseteq D$, if $\mathcal{K}$ is in DL-Lite$_{horn}^{\mathcal{H}}$.* □

Lastly, we consider so-called *simulations*, which in [Baa03] are described as binary relations between nodes of two so-called $\mathcal{EL}$ description graphs that respect the labels and edges of those graphs. Such an $\mathcal{EL}$ description graph is obtained for an interpretation $\mathcal{I}$ by regarding $\mathcal{I}$ as a graph such that the domain elements are the nodes, labeled by the concept names the elements satisfy; and the (labeled) edges are given by the roles connecting the elements in $\mathcal{I}$. We define the notion of simulation directly w.r.t. two interpretations.

Definition 2.16 A relation $\sigma \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{J}$ is a *simulation* (of $\mathcal{I}$ by $\mathcal{J}$), written $\sigma \colon \mathcal{I} \to \mathcal{J}$, iff the following hold for all $(d, e) \in \sigma$:

- $d \in A^\mathcal{I}$ implies $e \in A^\mathcal{J}$ for all $A \in \mathsf{N_C}$, and
- $(d, d') \in R^\mathcal{I}$ implies that there is an element $e' \in \Delta^\mathcal{J}$ such that $(d', e') \in \sigma$ and $(e, e') \in R^\mathcal{J}$ for all $R \in \mathsf{N_R}$. ♢

It is easy to inductively construct a simulation of the finite canonical interpretation of a KB $\mathcal{K}$ by any other model of $\mathcal{K}$.

Lemma 2.17 *For every model $\mathcal{J}$ of an $\mathcal{EL}$ knowledge base $\mathcal{K}$, there is a simulation σ of $\mathcal{I}_\mathcal{K}^f$ by $\mathcal{J}$ such that $(a, a^\mathcal{J}) \in \sigma$ for all $a \in \mathsf{N_I}(\mathcal{K})$.* □

2.2 Propositional Linear Temporal Logic

Propositional linear temporal logic (LTL[5]), also known as *propositional temporal logic*, extends propositional logic with modal operators to represent past and future moments in formulas. Accordingly, the signatures are as in propositional logic, sets of propositional variables.

Definition 2.18 (Syntax of Propositional LTL) Let $\mathcal{P} = \{p_1, \ldots, p_\ell\}$ be a finite propositional logic signature. The set of *propositional LTL formulas* over $\mathcal{P}$ is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi_1 \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc_F \varphi_1 \mid \bigcirc_P \varphi_1 \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \mid \varphi_1 \,\mathcal{S}\, \varphi_2$$

where $p \in \mathcal{P}$, and φ_1 and φ_2 are formulas, in their turn. ♢

Observe that LTL allows for Boolean negation and conjunction. The operators $\bigcirc_F$ and $\bigcirc_P$ are called "next" and "previous", respectively. The formula $\varphi_1 \,\mathcal{U}\, \varphi_2$ stands for "φ_1 until φ_2", and $\varphi_1 \,\mathcal{S}\, \varphi_2$ represents its dual, read "φ_1 since φ_2". We may use $\bigcirc_F^i$ to denote a sequence of i $\bigcirc_F$-operators, and similar for the previous operator. We assume

[5] For simplicity, we often drop the prefix "propositional" and simply refer to linear temporal logic (LTL) throughout the work.

Operator	Definition	Name
$\varphi_1 \vee \varphi_2$	$\neg(\neg\varphi_1 \wedge \neg\varphi_2)$	disjunction
$\varphi_1 \rightarrow \varphi_2$	$\neg(\varphi_1 \wedge \neg\varphi_2)$	implication
$\Diamond_F\varphi$	$\mathit{true}\,\mathcal{U}\,\varphi$	eventually (some time in the future)
$\Box_F\varphi$	$\neg\Diamond_F\neg\varphi$	always in the future
$\Diamond_P\varphi$	$\mathit{true}\,\mathcal{S}\,\varphi$	historically (some time in the past)
$\Box_P\varphi$	$\neg\Diamond_P\neg\varphi$	always in the past

Figure 2.2: Definitions of derived operators for propositional LTL.

true to denote an arbitrary but fixed propositional tautology (e.g., $p \vee \neg p$, where p is a propositional variable) and *false* to denote its negation. As usual, further derived operators can be defined as in Figure 2.2. We further use $\varphi_1 \leftrightarrow \varphi_2$ as an abbreviation for $(\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_1 \leftarrow \varphi_2)$.

The operators $\bigcirc_F$ and $\mathcal{U}$ are called the *future operators*, and $\bigcirc_P$ and $\mathcal{S}$ are the *past operators*. Together, they represent the *temporal operators*. In accordance with that, a propositional LTL formula is called a *future formula* if it contains no past operators and a *past formula* if it contains no future operators. An LTL formula is called *separated* if no future operators occur in the scope of past operators and vice versa. A subformula of a separated LTL formula φ is a *top-level future (past) formula* of φ if it is of one of the forms in 2.1 (2.2) and occurs in φ at least once in the scope of no other temporal operator:

$$\bigcirc_F \varphi_1, \neg(\bigcirc_F\varphi_1), \varphi_1\,\mathcal{U}\,\varphi_2, \neg(\varphi_1\,\mathcal{U}\,\varphi_2) \tag{2.1}$$

$$\bigcirc_P \varphi_1, \neg(\bigcirc_P\varphi_1), \varphi_1\,\mathcal{S}\,\varphi_2, \neg(\varphi_1\,\mathcal{S}\,\varphi_2) \tag{2.2}$$

The set of all subformulas of an LTL formula φ is denoted by $\mathbb{S}(\varphi)$.[6] The set $\mathrm{Clo}(\mathcal{F})$ denotes the closure under negation of $\bigcup_{\varphi\in\mathcal{F}} \mathbb{S}(\varphi)$.

In propositional LTL, the flow of time is considered to be bounded with respect to the past, discrete, and, as the name suggests, linear. It is represented by the sequence of natural numbers, such that every *point in time* (also *time point* or *moment*) is represented by one number. An LTL interpretation then is a corresponding *structure*: a sequence of propositional interpretations which, respectively, determine the propositions that are true at the corresponding points in time.

Definition 2.19 (Semantics of Propositional LTL) Let $\mathcal{P} = \{p_1, \ldots, p_\ell\}$ be a finite propositional logic signature. A *propositional LTL structure* for $\mathcal{P}$ is an infinite sequence $\mathfrak{W} = (w_i)_{i\geq 0}$ of *worlds* $w_i \subseteq \mathcal{P}$.

A propositional LTL structure $\mathfrak{W} = (w_i)_{i\geq 0}$ *satisfies* a propositional LTL formula φ at (time point) $i \geq 0$, written $\mathfrak{W}, i \models \varphi$, if the corresponding condition of Figure 2.3 is satisfied. The fact that $\mathfrak{W}, i \models \varphi$ does not hold is denoted by $\mathfrak{W}, i \not\models \varphi$.

If $\mathfrak{W}, 0 \models \varphi$, then $\mathfrak{W}$ is a *model* of φ.

A propositional LTL-formula φ is *satisfiable* if it has a model. Two propositional LTL formulas φ_1 and φ_2 are *equivalent*, written $\varphi_1 \equiv \varphi_2$, if they have the same models. $\Diamond$

[6] Recall that the same notation is used to denote the set of subconcepts of a concept.

Formula φ	Condition for $\mathfrak{W}, i \models \varphi$
p	$p \in w_i$
$\neg\varphi_1$	$\mathfrak{W}, i \not\models \varphi_1$
$\varphi_1 \wedge \varphi_2$	$\mathfrak{W}, i \models \varphi_1$ and $\mathfrak{W}, i \models \varphi_2$
$\bigcirc_F \varphi_1$	$\mathfrak{W}, i+1 \models \varphi_1$
$\bigcirc_P \varphi_1$	$i > 0$ and $\mathfrak{W}, i-1 \models \varphi_1$
$\varphi_1 \,\mathcal{U}\, \varphi_2$	there is a $k \geq i$, such that $\mathfrak{W}, k \models \varphi_2$ and, for all j, $i \leq j < k$, we have $\mathfrak{W}, j \models \varphi_1$
$\varphi_1 \,\mathcal{S}\, \varphi_2$	there is a k, $0 \leq k \leq i$, such that $\mathfrak{W}, k \models \varphi_2$ and, for all j, $k < j \leq i$, we have $\mathfrak{W}, j \models \varphi_1$

Figure 2.3: Semantics of propositional LTL formulas for an interpretation $\mathfrak{W} = (w_i)_{i \geq 0}$ for a signature $\mathcal{P}$, assuming $p \in \mathcal{P}$.

The empty conjunction and disjunction are interpreted as *true* and *false*, respectively. Again, the signature is in the following generally not mentioned explicitly if it is irrelevant or clear from the context.

Above, the operators $\mathcal{U}$ and $\mathcal{S}$ are defined in their *non-strict* version. The semantics of the *strict* operators $\mathcal{U}^<$ and $\mathcal{S}^<$ differs in that the parameters j and k in Figure 2.19 must not equal i. This means that $\mathfrak{W}, i \models \varphi_2$, which implies $\mathfrak{W}, i \models \varphi_1 \,\mathcal{U}\, \varphi_2$, does not imply $\mathfrak{W}, i \models \varphi_1 \,\mathcal{U}^< \varphi_2$; and similar for the since operator. However, in the presence of the $\bigcirc_F$-operator, $\mathcal{U}$ and $\mathcal{U}^<$ can be expressed in terms of each other. Specifically, the formula $\varphi_1 \,\mathcal{U}\, \varphi_2$ is equivalent to $\varphi_2 \vee (\varphi_1 \,\mathcal{U}^<\, \varphi_2)$, and $\varphi_1 \,\mathcal{U}^<\, \varphi_2$ is equivalent to $\varphi_1 \wedge \bigcirc_F(\varphi_1 \,\mathcal{U}\, \varphi_2)$; and similar for $\mathcal{S}$ and $\mathcal{S}^<$.

Further, note that the above definition of propositional LTL extends the usual definition of that logic, which only considers the temporal operators $\bigcirc_F$ and $\mathcal{U}$ [Pnu77]. For that reason, this extended logic is often referred to as Past-LTL. An important result for this logic, the so-called *separation theorem* [Gab87, Thm. 2.4], is given below.

Lemma 2.20 ([Gab87, Thm. 2.4]) *Every propositional LTL formula φ is equivalent to a propositional LTL formula that is separated.* □

Note that [Gab87] actually consider a slightly different temporal logic allowing only $\mathcal{S}^<$ and $\mathcal{U}^<$ as temporal operators. However, it is well-known that $\bigcirc_F$ and $\bigcirc_P$ can be simulated in this setting: $\bigcirc_F \varphi \equiv \mathit{false}\,\mathcal{U}^<\, \varphi$ and $\bigcirc_P \varphi \equiv \mathit{false}\,\mathcal{S}^<\, \varphi$. Moreover, the non-strict versions of the operators can be expressed in terms of the strict ones and the other way around while retaining separation, as shown above. Thus, Lemma 2.20 holds also for the logic we focus on. The size of the resulting separated LTL formula then however may be non-elementary in the size of the original formula (i.e., specifically, the number of stacked exponents is determined by the number of alternations between past operators and future operators) [Gab87]. But we use this result in a context where the size of the formula is irrelevant.

We close this section on LTL by describing the procedure for deciding LTL satisfiability originally proposed in [SC85, Sec. 4] in Algorithm 2.1, which we extend in Chapters 4, 5, and 6 to obtain our results. In particular, [SC85] propose a nondeterministic algorithm which runs in an amount of space polynomially bounded by the

Algorithm 2.1: Procedure for Deciding LTL Satisfiability

Input: LTL formula φ
Output: *true* if φ is satisfiable, otherwise *false*

```
1  i := 0
2  s := Guess a number ≤ 2^|φ| and > 0
3  p := Guess a number ≤ 4^|φ|
4  F_next := ∅, F_s := ∅, F_U := ∅
5  F_pres := Guess a subset of Clo({φ})
6  if not CONSISTENT(F_pres) or not INITIAL(F_pres) or φ ∉ F_pres then
7      return false
8  while i ≤ s + p do
9      if i > 0 then F_pres := F_next
10     F_next := Guess a set of subformulas of φ
11     if not CONSISTENT(F_next) or not TCONSISTENT(F_pres, F_next) then
12         return false
13     if i = s then
14         F_s := F_pres
15         F_U := All formulas of the form φ1 U φ2 ∈ F_s
16     if i ≥ s then
17         F_U := All formulas of the form φ1 U φ2 ∈ F_U such that φ2 ∉ F_pres
18     i := i + 1
19 if F_U = ∅ and TCONSISTENT(F_pres, F_s) then
20     return true
21 return false
```

formula φ considered; we consider $|\varphi|$ to denote the number of symbols occurring in φ. The idea for constructing an LTL structure $\mathfrak{W}$ satisfying φ is to iteratively guess subsets of $\mathrm{Clo}(\{\varphi\})$ that represent the subformulas satisfied in $\mathfrak{W}$ at each point in time. More precisely, every such set induces a unique world containing exactly the propositional variables that are true in the guessed set. In what follows, we describe that procedure, given as Algorithm 2.1, in more detail. We thereby rely on the subprocedures below.

- `CONSISTENT`: Given a set $\mathcal{F}$ of LTL formulas, it checks the Boolean consistency of the latter by returning true iff the following hold for all $\varphi \in \mathrm{Clo}(\mathcal{F})$:
  - $\varphi = \varphi_1 \wedge \varphi_2 \in \mathcal{F}$ iff $\varphi_1, \varphi_2 \in \mathcal{F}$,
  - $\varphi = \neg\varphi_1 \in \mathcal{F}$ iff $\varphi_1 \notin \mathcal{F}$.
- `INITIAL`: Given a set $\mathcal{F}$ of LTL formulas, it checks if $\mathcal{F}$ can describe the formulas satisfied at time point 0 in an LTL structure by returning true iff the following hold for all $\varphi \in \mathrm{Clo}(\mathcal{F})$:
  - $\varphi = \varphi_1 \,\mathcal{S}\, \varphi_2 \in \mathcal{F}$ iff $\varphi_2 \in \mathcal{F}$
  - $\varphi = \bigcirc_P \varphi_1 \notin \mathcal{F}$.
- `TCONSISTENT`: Given two sets $\mathcal{F}_{pres}$ and $\mathcal{F}_{next}$ of LTL formulas, it checks if they can be satisfied in an LTL structure at consecutive time points by returning true iff the following hold for all $\varphi \in \mathrm{Clo}(\mathcal{F})$:
  - $\varphi = \bigcirc_F \varphi_1 \in \mathcal{F}_{pres}$ iff $\varphi_1 \in \mathcal{F}_{next}$,
  - $\varphi = \bigcirc_P \varphi_1 \in \mathcal{F}_{next}$ iff $\varphi_1 \in \mathcal{F}_{pres}$,
  - $\varphi = \varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathcal{F}_{pres}$ iff $\varphi_2 \in \mathcal{F}_{pres}$ or ($\varphi_1 \in \mathcal{F}_{pres}$ and $\varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathcal{F}_{next}$),
  - $\varphi = \varphi_1 \,\mathcal{S}\, \varphi_2 \in \mathcal{F}_{next}$ iff $\varphi_2 \in \mathcal{F}_{next}$ or ($\varphi_1 \in \mathcal{F}_{next}$ and $\varphi_1 \,\mathcal{S}\, \varphi_2 \in \mathcal{F}_{pres}$).

The procedure is based on the fact that, if φ is satisfiable, then there must be a *periodic* model of φ with a period that starts at a time point at most exponential in the size of φ and is of length at most exponential in the size of φ [SC85, Theorem 4.7]. This means that the algorithm can iterate over all time points until the end of the period using a counter i that can be represented in polynomial space. The start s and length p of the period are guessed in the beginning.[7]This approach works specifically because the algorithm does not store all the sets of subformulas guessed during the iteration. Instead, it focuses on only four such sets $\mathcal{F}_{pres}$, $\mathcal{F}_{next}$, $\mathcal{F}_s$, and $\mathcal{F}_{\mathcal{U}}$, which are updated during processing and occupy only a polynomial amount of memory. $\mathcal{F}_{pres}$ specifies $\mathfrak{W}$ w.r.t. the present time point i, $\mathcal{F}_{next}$ describes the world for the next time point, $\mathcal{F}_s$ the one for time point s, and $\mathcal{F}_{\mathcal{U}}$ is used as auxiliary set. $\mathcal{F}_{pres}$ is guessed in the beginning and it is checked if this set can describe the formulas satisfied at time point 0 in an LTL structure that is a model of φ. Subsequently, the counter is continuously incremented and, until it reaches the beginning of the period s, in each step: $\mathcal{F}_{pres}$ and $\mathcal{F}_{next}$ are updated, which means that a set of subformulas of φ is guessed for the latter; further, the Boolean consistency of the new set, and its consistency with the set for the present time point (according to the temporal operators) are checked. Regarding the latter,

[7]For simplicity, we additionally require $s > 0$ in Algorithm 2.1; this is clearly without loss of generality.

note that the satisfiability test for subformulas of the form $\varphi_1 \mathcal{U} \varphi_2$ may be deferred to the next iteration step if $\varphi_1 \in \mathcal{F}_{pres}$. At the beginning of the period, the current set $\mathcal{F}_{pres}$ is stored in $\mathcal{F}_s$ and, until the end of the period, the algorithm continues the iteration as before. In addition, it however has to consider the $\mathcal{U}$-subformulas deferred at time point s to make sure that they are satisfied within the period. If it has reached the end of the period, it checks if the latter is the case and if $\mathcal{F}_s$, guessed for describing $\mathfrak{W}$ at the beginning of the period, can indeed fulfill that function.

Note that [SC85] do not regard the $\bigcirc_P$-operator, which is considered by us. However, we can obviously assume that the sets of subformulas that are guessed also include subformulas that include this operator and adapt the tests correspondingly (see the above specifications of `INITIAL` and `TCONSISTENT`). In particular, this does not affect the space requirements of the algorithm because the period that has to be guessed is still exponential in the size of the considered formula. Furthermore, we present the algorithm adapted to our setting, where satisfiability is to be decided w.r.t. time point 0. The original algorithm checks satisfiability at a time point given as argument.

We recapitulate the correctness and space complexity of the procedure.

Lemma 2.21 ([SC85, Thm. 4.1 and 4.7]) *Algorithm 2.1 nondeterministically decides the satisfiability of a given LTL formula φ by using an amount of space that is polynomially bounded in $|\varphi|$.* □

2.3 Computational Complexity

Computational complexity theory studies the inherent difficulty of computational problems and classifies them accordingly. Though large parts of this work rely on results from this field of computer science, a general introduction to it is beyond the scope of this work; we refer the reader to [AB09] for a detailed overview. This section is dedicated to basics that are important for this work.

In particular, we study the complexity of *decision problems*, which are problems that can be answered by either "yes" or "no", such as the question if $\langle \mathcal{O}, \mathcal{A} \rangle \models \varphi$ holds for an ontology $\mathcal{O}$, an ABox $\mathcal{A}$, and a CQ φ. In that context, mathematical models of computation (e.g., Turing machines (TMs)) serve as a means to quantify the amount of resources (i.e., usually time and space, representing storage) needed to solve the problems, in dependence of the size of the input problem. Together, the considered kind of problem and model of computation, the resources in focus, and the specific bounds placed on the resources characterize a complexity class.

Orthogonally to the complexity classes, we discern *combined complexity* and *data complexity*, depending on whether the problem size is determined by all of the input or only the data (i.e., in the example, the assertions in $\mathcal{A}$). In this context, we assume the *size* of an ontology $\mathcal{O}$, written $|\mathcal{O}|$, to be the number of symbols that is required to write $\mathcal{O}$ down in the alphabet provided by its signature together with the necessary auxiliary symbols; the size of an ABox (query) is defined correspondingly.[8] For the above example, the size of the input would then be $|\mathcal{O}| + |\mathcal{A}| + |\varphi|$ and $|\mathcal{A}|$ regarding combined

[8] Although we sometimes regard CQs φ as sets of atoms (see Section 2.1.2), $|\varphi|$ denotes the number of symbols that is required to write down φ as described throughout the thesis (i.e., instead of the number of atoms in the CQ).

and data complexity, respectively. Note that data complexity is of special interest if the size of the data is considerably bigger than the rest of the input. In that case, the complexity may drop considerably if input different from the data is disregarded. For instance, the CQ entailment problem in *DL-Lite*$_{core}$ depends polynomially on both the theory and the query, but only in a logarithmic factor on the data, considering (deterministic) TMs and computation time. Hence, if the size of both the ontology and the query is negligible, a classification as in LogTime w.r.t. data complexity better describes the dependence of the computation time on the input than a classification of in P w.r.t. combined complexity.

We primarily consider Turing machines, which are the classical computation model since they capture the intuitive notion of an algorithm. TMs can be further specified as being, for example, deterministic, non-deterministic, or alternating; however, though some more details on the latter kind are given below, we again refer to [AB09] for formal definitions of the machines and the associated complexity classes. In this work, we consider classes between LogTime and 2-ExpTime. Regarding a specific bound, the nondeterministic version of a class—prefixed by the letter N—subsumes the deterministic version, the space classes subsume those associated to time, and are subsumed by exponentially larger time classes. For example, regarding the polynomial bound, the following inclusions hold:

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSpace} \subseteq \mathrm{NPSpace} \subseteq \mathrm{ExpTime}.$$

Recall that the LTL satisfiability problem is in PSpace [SC85, Thm. 4.1] (see Algorithm 2.1). The equality PSpace = NPSpace was shown by Savitch [Sav70, Thm. 1] and rather important for our PSpace results. In most of the other cases, the exact relationships between the classes are however still unknown. Note that the problems contained in P are also called *tractable* because they are assumed to be efficiently solvable in practice.

Since some of the problems studied in this work are of very low data complexity, we consider *Boolean circuits*, a second, but less common, computation model.[9] Circuits model hardware and roughly formalize the familiar "silicon chip". Circuit complexity classes therefore characterize problems that can be efficiently solved on highly parallel computers [AB09, Thm. 6.27]. In what follows, we give a rather informal overview of this computation model. A *Boolean circuit* is a directed acyclic graph with $n \in \mathbb{N}$ input nodes (i.e., vertices with no incoming edges) and one output node (i.e., a vertex with no outgoing edges). All vertices that are no input nodes are *gates* labeled by one of AND, OR, and NOT. Size (i.e., the number of gates, representing processors) and depth of a circuit (i.e., the length of the longest directed path from an input to the output node) represent the space and time bounds, respectively. In this work, we regard the class NC^1, which captures all problems computable by circuits of size polynomial in n and depth $\mathcal{O}(\log n)$, where n represents the size of the problem. Note that the rather prominent class AC^0 is defined correspondingly, except that the depth must be constant ($\mathcal{O}(\log^0 n)$) and the gates may have unbounded fan-in (i.e., the OR and AND gates can be applied to more than two bits).

[9] Turing machine classes below LogTime are not commonly used.

To obtain a worst-case complexity C for a problem P, we first require it to be *hard* for the complexity class (under a given type of reduction). This is the case if any problem in C can be solved by reducing it to P, and the reduction is "significantly easier" than solving the problem directly. Moreover, P is *complete* for the class (for that type of reduction) if it is *hard* for that class and *contained* in it. For instance, [SC85, Thm. 4.1] actually establish PSPACE-completeness of the LTL satisfiability problem. In this work, we regard reductions in P if not stated otherwise.

For obtaining completeness results, comparisons between complexity classes for different kinds of computation models may thus be useful. However, unlike TMs where one machine is in focus, circuits are *non-uniform* models of computation, which means that instances of the same problem that are of different size are processed by different circuits. For that reason, a problem is associated with a set (also *family*) of circuits, each of which is dedicated to a problem instance of specific size $n \in \mathbb{N}$. The classes thus may contain problems that cannot be decided algorithmically and, hence, are not contained in any TM class (e.g., any problem can be decided by a circuit family where each circuit is of size $O(n * 2^n)$, by encoding the decision for every input). For this reason, a specific uniformity condition is often imposed on the circuit families; such a condition restricts the class to problems whose associated circuits (i.e., a description of them) can be computed by a particular TM, given the size of the input. We apply hardness results for uniform circuit complexity classes to obtain hardness results for TM complexity classes and rely on the result that LOGTIME-uniform NC^1 equals ALOGTIME [BIS90, Lem. 7.2], the class of problems solvable in logarithmic time with an alternating TM. Observe that, to obtain such a result, the reduction must be chosen according to the uniformity of the class it is used to show hardness for (e.g., to show that a problem is hard for LOGTIME-uniform NC^1, we can use a LOGTIME-uniform AC^0 reduction;[10] in contrast, a nonuniform AC^0 reduction would not fit).

The uniform version of AC^0 is of special importance because it also equals FO, the class of problems that can be described in first-order logic [BIS90, Thm. 9.1]. FO is a descriptive complexity class; such a class is characterized by the logic needed to express the languages (i.e., the problems) in them.

First-Order Rewritability

We specify the notion of *first-order (FO) rewritability* in a rather general way.

Definition 2.22 (first-order rewritable) A decision problem is *first-order rewritable* if there exists a first-order formula φ such that, for every instance P of the problem, we can effectively construct a first-order structure $\mathcal{I}_P$ that is solely based on the problem input and such that φ is satisfied in $\mathcal{I}_P$ iff the answer to P is "yes". ♢

For query answering problems regarding *DL-Lite*, research often targets containment in AC^0 if the focus is on data complexity. That is, the FO formula then must not depend on the data and is considered to be efficiently encodable and evaluable over it, by using

[10] Note that AC^0 is a true subset of NC^1 since unbounded (but polynomial in n) fan-in can be simulated using a tree of ORs/ANDs of depth $\mathcal{O}(\log n)$[AB09, p. 118]; and the problem of deciding if a given word is in the language PARITY $:= \{x : x \text{ has an odd number of 1s}\}$ is not in AC^0 [FSS84, Lemma 4.2, Theorem 4.3] but in NC^1 (see Example 6.26 in [AB09]).

standard database management systems. This is particularly the case because the shape of the ABoxes $\mathcal{A}$ containing the data allows to define FO structures $\mathrm{DB}(\mathcal{A})$ of the form of finite databases that, at the same time, represent minimal models of the respective ABoxes.

Definition 2.23 ($\mathrm{DB}(\mathcal{A})$) For a given *DL-Lite* ABox $\mathcal{A}$, the first-order structure $\mathrm{DB}(\mathcal{A}) = (\mathsf{N_I}(\mathcal{A}), \cdot^{\mathrm{DB}})$ over the *individual domain* $\mathsf{N_I}(\mathcal{A})$ contains the following relations for all $B \in \mathbb{B}(\mathcal{A})$ and $R \in \mathsf{N_R}(\mathcal{A})$:[11]

$$\mathsf{B}^{\mathrm{DB}} := \{(a) \mid B(a) \in \mathcal{A}\},$$
$$\mathsf{R}^{\mathrm{DB}} := \{(a,b) \mid R(a,b) \in \mathcal{A}\}. \qquad \diamondsuit$$

The semantics of the satisfaction relation $\models$ is specified as usual:

$$\mathrm{DB}(\mathcal{A}) \models \mathsf{B}(a) \quad \text{iff} \quad (a) \in \mathsf{B}^{\mathrm{DB}},$$
$$\mathrm{DB}(\mathcal{A}) \models \mathsf{R}(a,b) \quad \text{iff} \quad (a,b) \in \mathsf{R}^{\mathrm{DB}}.$$

Note that, if we use this structure with FO formulas encoding ontological knowledge (i.e., the problem input in ontology-based decision problems) as described above, then we assume that the database contains corresponding relations for all basic concepts and role names occurring in the ontology. Furthermore, observe that every real relational database that contains only unary and binary relations can be regarded as such a structure.

With this definition of the database we follow the approach of [Cal+05]. Instead of basic concepts, [Cal+07b; BAC10] consider only concept names as relations. Note that this makes no difference in our context, so that we can rely on results of [BAC10], assuming their FO rewritings to be adapted correspondingly. In particular, we apply the FO *rewritings* of KB inconsistency and UCQ answering proposed in that paper (see esp. Theorems 5 and 6 and Lemmas 10–12 in [BAC10]).

Lemma 2.24 ([BAC10]) *For a KB $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ in DL-Lite$_{horn}^{\mathcal{H}}$ and Boolean UCQ φ, we have the following:*

- *$\mathcal{K}$ is inconsistent iff $DB(\mathcal{A}) \models q_{unsat(\mathcal{O})}$.*
- *$\langle \mathcal{O}, \mathcal{A} \rangle \models \varphi$ iff $DB(\mathcal{A}) \models \mathsf{PerfectRef}(\varphi, \mathcal{O})$.* $\square$

We conclude this section by specifying alternating TMs, which occur in both hardness and containment proofs in this work.

Alternating Turing Machines

Alternating Turing machines extend nondeterministic Turing machines by labeling all states as either *existential* or *universal*. The former correspond to states in nondeterministic machines. If the machine is in a universal state, then all transitions that apply have to lead to an accepting state for the current run to be successful.[12]

[11] Note that, in later chapters, where we refer to this definition, we allow assertions of basic concepts to occur in ABoxes.

[12] The notions *configuration*, *transition*, and *run* are defined similarly as with nondeterministic TMs; note however that a run of an alternating TM is a tree.

Definition 2.25 An *alternating Turing machine* (ATM) $\mathfrak{M} = (\mathcal{Q}, \Sigma, \Gamma, q_0, \Delta)$ is specified as follows:

- $\mathcal{Q} = \mathcal{Q}_\exists \cup \mathcal{Q}_\forall$ is a finite set of states partitioned into *existential states* $\mathcal{Q}_\exists$ and *universal states* $\mathcal{Q}_\forall$;
- Σ is the *input alphabet*;
- Γ is the set of *working symbols* containing a *blank symbol* B and all symbols from Σ;
- $q_0 \in \mathcal{Q}_\exists \cup \mathcal{Q}_\forall$ is the *initial state*;
- Δ denotes the *transition relation*, for which we have

$$\Delta \subseteq \mathcal{Q} \times \Gamma \times \mathcal{Q} \times \Gamma \setminus \{B\} \times \{L, R, N\}.$$

A *step* of $\mathfrak{M}$ consists of reading one symbol, writing a symbol, moving the head left or right one tape cell, and entering a new state, in accordance with the transition relation.

A *configuration* of $\mathfrak{M}$ is an element of $\Gamma^* \mathcal{Q} \Gamma^*$. A configuration α' is a *successor* of a configuration α, written $\alpha \vdash \alpha'$, if α' follows from α in one step according to the transitions in Δ. The reflexive transitive closure of $\vdash$ is denoted by $\vdash^*$. A configuration α is *accepting* (vs. *rejecting*) iff

- α is a universal configuration and all its successor configurations are accepting, or
- α is an existential configuration and at least one of its successor configurations is accepting;

α is a *halting* configuration if it has no successor configurations.

A *computation* of $\mathfrak{M}$ on a word $w \in \Sigma^*$ is a sequence $\alpha_0 \vdash \alpha_1 \vdash \cdots \vdash \alpha_n$ of successive configurations starting with $\alpha_0 = q_0 w$, the *initial* configuration.

$\mathfrak{M}$ *accepts* (vs. *rejects*) a word $w \in \Sigma^*$ iff the configuration $q_0 w$ is accepting. $\mathfrak{M}$ accepts w *in time* t if $\mathfrak{M}$ accepts w and there is no computation with more than t steps. $\mathfrak{M}$ accepts w *in space* s if $\mathfrak{M}$ accepts w and all configurations α reachable from $q_0 w$ take at most s space; that is, $|\alpha| \leq s$. ♢

Note that a configuration without successor is accepting iff it is universal. We write $\Delta(q, \sigma)$ to denote the set $\{(p, \varrho, M) \mid (q, \sigma, p, \varrho, M) \in \Delta\}$.

3 Introduction to Temporal Query Answering

In this chapter, we introduce the query languages and the problems Chapters 4 to 7 focus on in detail, outline some of our solutions, and give a general overview of related work on temporal query answering in DLs. In particular, we focus on a generic DL $\mathcal{DL}$ and (atemporal) query language $\mathcal{QL}$, instantiated later in this chapter; regard *temporal* $\mathcal{QL}$ *queries* (TQs) (or *temporal queries*, if $\mathcal{QL}$ is irrelevant or clear from the context); and introduce the reasoning problems of *TQ satisfiability* and *TQ entailment* w.r.t. a temporal knowledge base written in $\mathcal{DL}$; note that *TQ answering* w.r.t. temporal knowledge bases—not necessarily formulated in a description logic—is studied in Chapter 8. The temporal queries are basically formulas of propositional LTL, but the propositions are replaced by $\mathcal{QL}$ queries; and the semantics is suitably lifted from propositional worlds to DL interpretations. That is, both the $\mathcal{QL}$ queries and the axioms from the temporal knowledge base are interpreted in DL interpretations, and the semantics is based on infinite sequences of such interpretations over the same non-empty domain, called *DL-LTL structures*.

In Section 3.1, we specify the syntax and semantics of the queries and define the problems. A general approach for obtaining containment results for them which has been proposed in the literature [BGL12; BBL13] is detailed thereafter in Section 3.2 since we apply it throughout this work. In Section 3.3, we then describe why this approach does not directly yield useful containment results for temporal query answering in lightweight description logics and specify problems to solve, our approach, and questions we investigate in the following chapters. In the remainder of the chapter, Section 3.4, we describe related work.

3.1 Temporal Queries

In this section, we specify the temporal query answering setting described in Chapter 1. In particular, the specification is similarly generic, we focus on *temporal knowledge bases* in a description logic $\mathcal{DL}$ (i.e., instead of in an arbitrary logic, which is considered in Chapter 8). In particular, we assume that a subset of the concept and role names is designated as being *rigid* (vs. *flexible*).[1] As outlined in Section 1.3, the intuition is that the interpretation of rigid names does not change over time. Specifically, the individual names are implicitly assumed to be rigid (i.e., in a DL-LTL structure, an individual name is interpreted by the same domain element at all time points). If a concept (axiom) contains only rigid symbols, then we may call it a *rigid* concept (axiom). We denote by $\mathsf{N_{RC}} \subseteq \mathsf{N_C}$ the rigid concept names, by $\mathsf{N_{RR}} \subseteq \mathsf{N_R}$ the rigid role names, and assume a DL signature to be of the form $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_{RC}}, \mathsf{N_R}, \mathsf{N_{RR}})$, in the following.

[1] In the literature, rigid and flexible symbols are also called *global* and *local*, respectively [Art+07].

Definition 3.1 (DL-LTL structure) An infinite sequence $\mathfrak{I} = (\mathcal{I}_i)_{i\geq 0}$ of interpretations $\mathcal{I}_i = (\Delta, \cdot^{\mathcal{I}_i})$ for a description logic signature $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_{RC}}, \mathsf{N_R}, \mathsf{N_{RR}})$ is a *DL-LTL structure* if it *respects rigid names*; that is:

$$X^{\mathcal{I}_i} = X^{\mathcal{I}_j} \text{ for all } X \in \mathsf{N_I} \cup \mathsf{N_{RC}} \cup \mathsf{N_{RR}} \text{ and } i, j \geq 0. \qquad \diamondsuit$$

As mentioned above, the interpretations in a DL-LTL structure share one domain (constant domain assumption). Further, note that we often do not explicitly mention the signature if it is irrelevant or clear from the context.

We may also use this terminology in other settings in that we consider interpretations $\mathcal{I}_1, \ldots, \mathcal{I}_\ell$ to *respect rigid names* if they agree on the interpretation of all rigid symbols.

Note that we employ a semantics that is nowadays standard in the field, as remarked in [Art+07, p. 1]:

> *"it is generally agreed that the semantics of combined temporal description logics should be based on the Cartesian products of the flow of time (...) and the domains of the DL interpretations. (...) This semantics corresponds to the semantics of first-order temporal logics (...). In fact, the translation of standard DLs into first-order logic can be extended to a translation of temporalised DLs into first-order temporal logics."*

Yet, observe that the usual approach of reducing reasoning in the setting with constant domains to reasoning with expanding, decreasing, or varying domains as, for example, detailed in [LWZ08, Sec. 3] does not work for lightweight DLs. In particular, the exact relations between the settings are not clear. Note that we discuss alternative temporal semantics w.r.t. applications at the end of Section 8.2.

Definition 3.2 (Syntax of Temporal Knowledge Bases) A *temporal knowledge base* (TKB) $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0\leq i\leq n} \rangle$ consists of an ontology $\mathcal{O}$ written in $\mathcal{DL}$ and a non-empty, finite *sequence of ABoxes* of *length* $n+1$, where the ABoxes contain only assertions of concept and role names. $\diamondsuit$

As with atemporal KBs, we assume all concept and role names occurring in some ABox of a TKB to also occur in its ontology.

We use the notation $\mathsf{N_{RC}}(\mathcal{O})$ for the set of all rigid concept names that occur in an ontology $\mathcal{O}$, and $\mathbb{B}_{\mathsf{R}}(\mathcal{O})$ and $\mathbb{B}^{\neg}_{\mathsf{R}}(\mathcal{O})$ for the restriction of $\mathbb{B}(\mathcal{O})$ and, respectively, $\mathbb{B}^{\neg}(\mathcal{O})$ to rigid concepts. $\mathsf{N_I}(\mathcal{K})$ denotes the set of all individual names occurring in the TKB $\mathcal{K}$, and $\mathbb{B}(\mathfrak{A})$ and $\mathsf{N_R}(\mathfrak{A})$ designate the basic concepts and role names in the ABox sequence $\mathfrak{A}$. For simplicity, we sometimes omit the brackets around the ABox sequence.

Definition 3.3 (Semantics of Temporal Knowledge Bases) A DL-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i\geq 0}$ over a domain Δ is a *model* of a TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0\leq i\leq n} \rangle$, written $\mathfrak{I} \models \mathcal{K}$, if the following hold:

- $\mathcal{I}_i \models \mathcal{O}$ for all $i \geq 0$,
- $\mathcal{I}_i \models \mathcal{A}_i$ for all $i \in [0, n]$.

A TKB is *consistent* (or *satisfiable*) if it has a model, and it is *inconsistent* (or *unsatisfiable*) otherwise. $\diamondsuit$

We denote the fact that $\mathfrak{I} \models \mathcal{K}$ does not hold by $\mathfrak{I} \not\models \mathcal{K}$.

As outlined above, TQs combine $\mathcal{QL}$ queries via LTL operators.

Definition 3.4 (Syntax of Temporal Queries) Let $\Sigma = (\mathsf{N_I}, \mathsf{N_C}, \mathsf{N_{RC}}, \mathsf{N_R}, \mathsf{N_{RR}})$ be a DL signature. The set of *temporal $\mathcal{QL}$ queries* (TQs) over Σ is defined by the following grammar:

$$\Phi ::= \varphi \mid \neg\Phi_1 \mid \Phi_1 \wedge \Phi_2 \mid \bigcirc_F \Phi_1 \mid \bigcirc_P \Phi_1 \mid \Phi_1 \,\mathcal{U}\, \Phi_2 \mid \Phi_1 \,\mathcal{S}\, \Phi_2$$

where φ is a $\mathcal{QL}$ query over Σ, and Φ_1 and Φ_2 are TQs, in their turn.

A TQ Φ is a $\mathcal{QL}$ *query literal* if it is of the form $(\neg)\varphi$ with φ being a $\mathcal{QL}$ query. It is *positive* if $\Phi = \varphi$ and otherwise *negative*. ♢

We denote the set of individual names occurring in a TQ Φ by $\mathsf{N_I}(\Phi)$. A TQ Φ *contains* a symbol X if X occurs in Φ, and Φ *contains* a $\mathcal{QL}$ query φ if φ occurs in Φ at least once not as part of another $\mathcal{QL}$ query occurring in Φ.

Observe that the definition of TQs based on $\mathcal{QL}$ queries is analogous to the definition of propositional LTL formulas based on propositions (see Definition 2.18). We hence can analogously use abbreviations *true*[2] and *false* and may apply derived operators corresponding to those in Figure 2.2. The empty conjunction and disjunction are interpreted as *true* and *false*, respectively.

The semantics of TQs is based on those of $\mathcal{QL}$ queries, which we assume to be based on interpretations and to be defined already. More precisely, we denote the fact that an interpretation $\mathcal{I}$ satisfies a $\mathcal{QL}$ query φ by $\mathcal{I} \models \varphi$.

Definition 3.5 (Semantics of Temporal Queries) For a given DL-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i \geq 0}$, an $i \geq 0$, and a TQ Φ, the satisfaction relation $\mathfrak{I}, i \models \Phi$ is defined by induction on the structure of Φ: for a $\mathcal{QL}$ query φ, $\mathfrak{I}, i \models \varphi$ holds if $\mathcal{I}_i \models \varphi$; for other kinds of TQs, the corresponding condition of Figure 2.3 has to be satisfied. $\mathfrak{I}$ is a *model* of Φ w.r.t. a TKB $\mathcal{K}$ if $\mathfrak{I} \models \mathcal{K}$ and $\mathfrak{I}, n \models \Phi$; and $\mathfrak{I}$ is a *model* of Φ if it is a model of Φ w.r.t. $\langle \emptyset, \emptyset \rangle$.

A Boolean TQ Φ is *satisfiable* (w.r.t. a TKB $\mathcal{K}$) if it has a model (w.r.t. $\mathcal{K}$); and Φ is *entailed* by a TKB $\mathcal{K}$, written $\mathcal{K} \models \Phi$, if every model of $\mathcal{K}$ is also a model of Φ w.r.t. $\mathcal{K}$. ♢

We denote the fact that $\mathfrak{I}, i \models \Phi$ and $\mathcal{K} \models \Phi$ do not hold by $\mathfrak{I}, i \not\models \Phi$ and $\mathcal{K} \not\models \Phi$.

Observe that a model of a TQ must satisfy the query at the current time point n, which differs from the corresponding definition for propositional LTL if $n > 0$. Moreover, for reasoning, we often consider TQs in the context of a TKB.

Definition 3.6 (Reasoning Problems for TQs) For a given Boolean TQ Φ and a TKB $\mathcal{K}$, there are the following reasoning problems:

- *TQ Satisfiability:* Is Φ satisfiable w.r.t. $\mathcal{K}$?
- *TQ Entailment:* Does $\mathcal{K} \models \Phi$ hold? ♢

We often solve the entailment problem by reducing it to the satisfiability problem: If $\neg\Phi$ is satisfiable, then Φ is not entailed.

[2] For instance, *true* may denote a fixed TQ $\varphi \vee \neg\varphi$, where φ is an arbitrary $\mathcal{QL}$ query.

According to [BBL15b], the entailment problem of a TQ Φ w.r.t. a TKB $\langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ can be solved by considering an extension of Φ w.r.t. $\langle \mathcal{O}, \emptyset \rangle$—regarding the trivial sequence of ABoxes.[3] Note that this sometimes eases analysis but is not useful if data complexity is considered, because the data is incorporated into the query.

Lemma 3.7 ([BBL15b, Lem. 6.1]) *Let $\mathcal{QL}$ be such that every assertion represents a $\mathcal{QL}$ query. For every TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ and Boolean TQ Φ, there is a Boolean TQ Φ' of size polynomial in the size of Φ and $\mathcal{K}$ such that $\mathcal{K} \models \Phi$ iff $\langle \mathcal{O}, \emptyset \rangle \models \Phi'$.* □

We often regard TQs Φ that do not contain temporal operators; for example, UCQs or conjunctions of CQ literals in case the $\mathcal{QL}$ queries are conjunctive queries. It can be readily checked that the satisfaction of Φ by a DL-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i \geq 0}$ at time point i then only depends on the interpretation $\mathcal{I}_i$, according to the definition and the conditions in Figure 2.3. For simplicity, we then often write $\mathcal{I}_i \models \Phi$ instead of $\mathfrak{I}, i \models \Phi$. In this context, it is also sufficient to consider classical knowledge bases $\langle \mathcal{O}, \mathcal{A} \rangle$, which can be regarded as TKBs with a single ABox.

Observe that TQs can be related to propositional LTL formulas in an intuitive way, by abstracting from the $\mathcal{QL}$ queries and considering propositions instead. This allows us to analyze the temporal structure of the TQ separately from the DL part.

Definition 3.8 (Propositional Abstraction) Let Φ be a TQ and $\mathcal{P}_\Phi$ be a finite set of propositional variables such that there is a bijection $\cdot^{\text{pa}}$ mapping the $\mathcal{QL}$ queries contained in Φ to elements of $\mathcal{P}_\Phi$.[4] The *propositional abstraction* Φ^{pa} of Φ w.r.t. $\cdot^{\text{pa}}$ is the propositional LTL formula obtained from Φ by replacing every $\mathcal{QL}$ query φ in Φ by φ^{pa}.

The *propositional abstraction* $\mathfrak{I}^{\text{pa}}$ of a DL-LTL structure $\mathfrak{I} = (\mathcal{I}_i)_{i \geq 0}$ w.r.t. $\cdot^{\text{pa}}$ is the propositional LTL structure $\mathfrak{I}^{\text{pa}} = (w_i)_{i \geq 0}$ where, for all $i \geq 0$,

$$w_i := \{\varphi^{\text{pa}} \mid \varphi \text{ is contained in } \Phi,\ \mathcal{I}_i \models \varphi\}. \qquad \diamondsuit$$

This abstraction obviously retains the semantics. That is, for a DL-LTL structure $\mathfrak{I}$ that is a model of a TKB $\mathcal{K}$ and a bijection $\cdot^{\text{pa}}$ as in Definition 3.8, we have: $\mathfrak{I}$ is a model of a TQ Φ w.r.t. $\mathcal{K}$ iff $\mathfrak{I}^{\text{pa}}$ is a model of Φ^{pa}.

In the remainder of the section, we present the two instantiations of TQs this work focuses on: temporalized DL axioms (investigated in Chapter 4) and temporal conjunctive queries (investigated in Chapters 5 to 7). Notes that, in Chapter 8, we consider various other instantiations.

Temporalized Description Logic Axioms

As it has been done for *TDL-Lite* [Art+07], *DL-Lite* logics that allow for temporal operators in concept expressions; $\mathcal{ALC}$ [BGL12]; and $\mathcal{SHOQ}$ [Lip14]; we combine the operators of LTL with $\mathcal{DL}$ axioms into the temporal query language $\mathcal{DL}$-LTL.[5] That

[3] The temporal query language considered in [BBL15b] is that of temporal conjunctive queries, but the result is independent of the $\mathcal{QL}$ queries and hence extends to our setting.

[4] Note that such a set $\mathcal{P}_\Phi$ and bijection $\cdot^{\text{pa}}$ obviously exist for every TQ Φ.

[5] Note that the queries in the related works do not contain past operators.

is, DL axioms are considered to be the $\mathcal{QL}$ queries. We specifically regard lightweight DLs $\mathcal{DL}$ and investigate the satisfiability problem.

In particular, we disregard TKBs by assuming them to be empty—which means that we always have $n = 0$. Observe that $\mathcal{DL}$-LTL then represents a kind of temporal DL, and TQs are also called *$\mathcal{DL}$-LTL formulas*. Nevertheless, there are close connections to the general setting.

Example 3.9 The question if an $\mathcal{DL}$ TKB $\langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ is consistent can also be decided by asking if the following $\mathcal{DL}$-LTL formula is satisfiable (see also Lemma 3.7):

$$\Box_F(\bigwedge \mathcal{O}) \wedge \bigwedge_{\substack{0 \leq i \leq n, \\ \alpha \in \mathcal{A}_i}} \bigcirc_F^i \alpha.$$

♢

Observe that the formula in the example is of special shape because it does not contain arbitrary combinations of CIs, but only a conjunction prefixed by $\Box_F$. This is known as $\mathcal{DL}$-LTL with *global CIs* (vs. *local* CIs) and investigated separately in this work.

Lastly, note that we generally use the notion *$\mathcal{DL}$ literals* for the literals mentioned in Definition 3.4 if $\mathcal{QL}$ is the language of $\mathcal{DL}$ axioms.

Temporal Conjunctive Queries

Temporal conjunctive queries (TCQs) are a temporal query language introduced in [BBL13]. They represent TQs where $\mathcal{QL}$ is the query language of conjunctive queries. [BBL13; BBL15b; BBL15a] investigate the TCQ entailment problem w.r.t. temporal knowledge bases in $\mathcal{ALC}$ and other expressive DLs, such as $\mathcal{SHOQ}$. Below, we describe how TCQ answering can be reduced to TCQ entailment. We investigate that problem regarding lightweight DLs.

We next introduce additional syntax, describe the relation between TCQ entailment and TCQ answering, and provide technical assumptions we make about TCQs throughout this thesis. For a TCQ Φ, we denote the set of variables occurring in Φ by $\mathsf{N_V}(\Phi)$; and the set of *free variables* of Φ, which is the union of the free variables of all CQs contained in Φ, by $\mathsf{N_{FV}}(\Phi)$. A TCQ Φ is a *Boolean* TCQ if $\mathsf{N_{FV}}(\Phi) = \emptyset$.

For determining complexity, we have to focus on a decision problem. The focus of research therefore is usually on query entailment instead of on query answering.

Definition 3.10 (TCQ Answering) For a given TCQ Φ, the problem of *TCQ Answering* is defined as follows: Determine all assignments $\mathfrak{a}$ (called *certain answers*) of the free variables of Φ to named individuals occurring in $\mathcal{K}$ such that $\mathcal{K} \models \mathfrak{a}(\Phi)$. ♢

As it is the case for UCQs, the set of certain answers to a Boolean TCQ is $\{()\}$ ("true") if $\mathcal{K} \models \Phi$ and $\emptyset$ ("false") otherwise. Query answering for Boolean TCQs thus represents the decision problem of query entailment. This is why we focus on Boolean TCQs in Chapters 5–7. Although answering non-Boolean queries is not necessarily efficient in practice, the problem of answering a TCQ Φ w.r.t. a TKB $\mathcal{K}$ can easily be reduced to TCQ entailment, by considering exponentially many entailment problems for the $|\mathsf{N_I}(\mathcal{K})|^{|\mathsf{N_{FV}}(\Phi)|}$ possible certain answers that are to be considered.

The technical assumptions we make about TCQs are as follows. We assume without loss of generality that the CQs contained in a Boolean TCQ Φ use disjoint variables

and denote by $\mathcal{Q}_\Phi$ the set of exactly those CQs.[6] We further assume that TCQs contain only individual names that occur in the ABoxes, and only concept and role names that occur in the ontology; this is clearly without loss of generality, especially because of the assumption that all concept and role names occurring in a TKB occur in its ontology. For simplicity, we assume all Boolean CQs contained in TCQs to be connected in the following; note that this assumption thus also applies to all (U)CQs we regard since they are also TCQs. To see that this is without loss of generality, consider a Boolean TCQ Φ containing a CQ φ that is not connected. In that case, φ can be replaced by a conjunction $\varphi_1 \wedge \cdots \wedge \varphi_\ell$ of CQs $\varphi_1, \dots, \varphi_\ell$ that represent a partition of φ. This conjunction is of linear size in the size of φ and the resulting TCQ has exactly the same models as Φ. More precisely, every homomorphism of φ into an interpretation $\mathcal{I}$ is also a homomorphism of, respectively, $\varphi_1, \dots, \varphi_\ell$ into $\mathcal{I}$ if it is restricted to the corresponding terms. And, because of the disjointness of $\mathsf{N_T}(\varphi_1), \dots, \mathsf{N_T}(\varphi_\ell)$, given by the definition of a partition, every collection of homomorphisms of $\varphi_1, \dots, \varphi_\ell$ into an interpretation $\mathcal{I}$ can be unified to a homomorphism of φ into $\mathcal{I}$.

3.2 A General Approach for Solving Satisfiability

In this section, we describe a general approach for solving the satisfiability problem for TQs which has been proposed in [BGL12; BBL15b] for $\mathcal{DL}$-LTL formulas and TCQs. We use this procedure to obtain several upper bounds. Recall that the TQ entailment problem, which we investigate in the context of TCQs, can be reduced to the satisfiability problem.

In a nutshell, the given satisfiability problem is reduced to two separate satisfiability problems—one in LTL and one in $\mathcal{DL}$. In what follows, we assume Φ to be a TQ, Φ^{pa} to be its propositional abstraction, and the corresponding bijection $\cdot^{\mathrm{pa}}$ to map the $\mathcal{QL}$ queries $\varphi_1, \dots, \varphi_m$ contained in Φ to propositions $p_1, \dots, p_m$ such that $p_i = \varphi_i^{\mathrm{pa}}$ for $i \in [1, m]$; we sometimes call φ_i *induced by* p_i. For a better understanding, we first disregard the TKB.

The goal is thus to find a model of Φ. For the LTL part, we therefore look for a model $\mathfrak{W}$ of Φ^{pa}. However, the DL part can obviously not be ignored entirely since not every model of Φ^{pa} is the propositional abstraction of a DL-LTL structure (e.g., the propositional abstraction of the $\mathcal{EL}$-LTL formula $\Phi = A \sqsubseteq B \wedge A(a) \wedge \neg B(a)$ is clearly satisfiable while Φ is not). We therefore collect the worlds occurring in $\mathfrak{W}$ in a (non-empty) set $\mathcal{W} \subseteq 2^{\{p_1, \dots, p_m\}}$ to later be able to check the DL part. This is captured by an LTL formula $\Phi^{\mathrm{pa}}_{\mathcal{W}}$ as follows:

$$\Phi^{\mathrm{pa}}_{\mathcal{W}} := \Phi^{\mathrm{pa}} \wedge \Box_F \left(\bigvee_{W \in \mathcal{W}} \left(\bigwedge_{p \in W} p \wedge \bigwedge_{p \in \overline{W}} \neg p \right) \right); \tag{3.1}$$

we use $\overline{W} := \{p_1, \dots, p_m\} \setminus W$ to denote the complement of a set $W \subseteq \{p_1, \dots, p_m\}$. Observe that the satisfiability of Φ implies the satisfiability of $\Phi^{\mathrm{pa}}_{\mathcal{W}}$ for some $\mathcal{W}$. This allows to proceed as follows: choose a set $\mathcal{W}$, test whether $\Phi^{\mathrm{pa}}_{\mathcal{W}}$ is satisfiable, and then

[6] If the variables were not disjoint, we could simply rename them. Note that this assumption also applies to Boolean UCQs.

check whether the model $\mathfrak{W}$ can indeed be the propositional abstraction of a DL-LTL structure.

To check the latter, we consider the conjunction $\bigwedge_{p_j \in W} \varphi_j$ for every $W \in \mathcal{W}$. However, the rigid names additionally make it necessary that these conjunctions are considered together and that we also consider the queries φ_j for which $p_j \in \overline{W}$ (e.g., the propositional abstraction of $\Phi = A(a) \wedge \bigcirc_F \neg A(a)$ is satisfiable while Φ is not if $A \in \mathsf{N_{RC}}$, since every DL-LTL structure respects rigid names). To not mix up the flexible names X occurring in different elements of $\mathcal{W}$, we introduce copies $X^{(i)}$ of them for all $i \in [1, |\mathcal{W}|]$; $X^{(i)}$ is called the *i-th copy* of X. In $\chi_{\mathcal{W}}$, we then use queries $\varphi_j^{(i)}$ for all $j \in [1, m]$, which are obtained from the corresponding queries φ_j by replacing every occurrence of a flexible name by its i-th copy:

$$\chi_{\mathcal{W}} := \bigwedge_{i=1}^{k} \Big(\bigwedge_{p_j \in W_i} \varphi_j^{(i)} \wedge \bigwedge_{p_j \in \overline{W_i}} \neg \varphi_j^{(i)} \Big). \tag{3.2}$$

[BGL12] apply this formula for characterizing the satisfiability problem in $\mathcal{ALC}$-LTL (see the claim in the proof of Lemma 4.3), and it can be easily shown that this result holds for TQs in general: The TQ Φ is satisfiable iff there is a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ such that $\Phi^{\mathrm{pa}}_{\mathcal{W}}$ and $\chi_{\mathcal{W}}$ are both satisfiable.

Regarding TCQs, [BBL15b] extend this approach to the setting with a TKB. We describe this procedure next, and consider a TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0 \le i \le n} \rangle$. As before, the task is to find a model of Φ and split into an LTL and a DL satisfiability problem. For the former, we similarly consider the set $\mathcal{W} = \{W_1, \dots, W_k\} \subseteq 2^{\{p_1,\dots,p_m\}}$ and LTL formula $\Phi^{\mathrm{pa}}_{\mathcal{W}}$. Yet, the two differences are now that the satisfiability is regarded at n—in line with the satisfiability problem for TQs w.r.t. a TKB which may contain data—and that, in addition to $\mathcal{W}$, we consider a second link to the DL part: a mapping $\iota\colon [0, n] \to [1, k]$ that maps time points to indexes from $\mathcal{W}$. This mapping points out the first $n + 1$ worlds, which have to be considered w.r.t. the respective ABoxes. The notion of *t-satisfiability* summarizes the LTL part.

Definition 3.11 (t-satisfiable) The LTL formula Φ^{pa} is *t-satisfiable* w.r.t. $\mathcal{W}$ and ι if there is an LTL structure $\mathfrak{W} = (w_i)_{i \ge 0}$ such that the following hold:

- $w_i \in \mathcal{W}$ for all $i \ge 0$,
- $\mathfrak{W}, n \models \Phi^{\mathrm{pa}}$,
- $w_i = W_{\iota(i)}$ for all $i \in [0, n]$. ♢

Note that the first condition captures the second conjunct of (3.1).

For the DL part, we can similarly consider the satisfiability of a TQ of the form (3.2) w.r.t. a TKB where the ontology contains copies of the CIs in $\mathcal{O}$ and the ABoxes are empty, if the TQ is extended such that it encodes the ABox assertions (see also Lemma 3.7). Observe, however, that this satisfiability test does not suit data complexity investigations, because the size of both the TQ and the ontology then depends on the data given. For that reason, [BBL15b] provide the following definition of *r-satisfiability* (r for "rigid") to summarize the DL part.

Definition 3.12 (r-satisfiable) The set $\mathcal{W}$ is *r-satisfiable* w.r.t. ι and $\mathcal{K}$ iff there are interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ as follows:

- the interpretations share the same domain and respect rigid names,
- the interpretations are models of $\mathcal{O}$,
- $\mathcal{J}_i$ is a model of $\chi_i := \bigwedge_{p_j \in W_i} \varphi_j \wedge \bigwedge_{p_j \in \overline{W_i}} \neg\varphi_j$ for all $i \in [1, k]$,
- $\mathcal{I}_i$ is a model of $\mathcal{A}_i$ and $\chi_{\iota(i)}$ for all $i \in [0, n]$. ♢

This definition explicitly asks for a shared domain and the consideration of rigid names. In addition, it states $k + n + 1$ TQ satisfiability problems: for the TQs χ_i w.r.t. $\langle \mathcal{O}, \emptyset \rangle$ where $i \in [1, k]$, and for the TQs $\chi_{\iota(i)}$ w.r.t. $\langle \mathcal{O}, \mathcal{A}_i \rangle$ where $i \in [0, n]$. Observe that these TQs do not contain temporal operators at all and that the TKBs are KBs as well.

The satisfiability of Φ w.r.t. $\mathcal{K}$ can then be decided by combining the two parts.

Lemma 3.13 ([BBL15b, Lem. 4.7]) *A TQ Φ has a model w.r.t. a TKB $\mathcal{K}$ iff there are a set $\mathcal{W} = \{W_1, \dots, W_k\} \subseteq 2^{\{p_1, \dots, p_m\}}$ and a mapping $\iota\colon [0, n] \to [1, k]$ such that*

- *Φ^{pa} is t-satisfiable w.r.t. $\mathcal{W}$ and ι, and*
- *$\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$.* □

The original proof of Lemma 3.13 in [BBL15b] considers TCQs and the DL $\mathcal{SHQ}$, but it is actually independent of both the DL and $\mathcal{QL}$ queries under consideration and hence also applies in our setting. In fact, regarding the empty TKB $\langle \emptyset, \emptyset \rangle$, the lemma is equivalent to the result from [BGL12] stated above. Specifically, the (trivial) mapping $\iota\colon [0, 0] \to [1, k]$ can be considered to be such that $\iota(0)$ points to the index $i \in [1, k]$ that exists due to the t-satisfiability requirement that $w_0 \in \mathcal{W}$, such that we have $w_0 = W_i$.

We lastly consider an alternative characterization of r-satisfiability. As outlined above, the latter can be decided using a TQ similar to (3.2) and an ontology $\mathcal{O}_{\mathcal{W},\iota}$. To this end, we consider *copies* $\alpha^{(i)}$ of ontology axioms α obtained from the axioms in $\mathcal{O}$ by replacing every occurrence of a flexible name by its i-th copy. In addition to the k worlds—or time points—already discerned, the first $n + 1$ time points have to be distinguished. This is basically because a set of assertions from some of the ABoxes may contradict a $\mathcal{QL}$ query not induced by some world, and because a $\mathcal{QL}$ query induced by a world may be contradicted by an assertion in one of the ABoxes (i.e., based on the assertion, the ontology may imply something contrary to what is stated in the $\mathcal{QL}$ query). Lemma 3.14 captures this approach.

Lemma 3.14 ([BBL15b, Lem. 4.14]) *Let $\mathcal{QL}$ be such that every assertion in $\mathcal{DL}$ is a $\mathcal{QL}$ query. The set $\mathcal{W}$ is* r-satisfiable *w.r.t. ι and $\mathcal{K}$ iff the conjunction $\chi_{\mathcal{W},\iota}$ of $\mathcal{QL}$ query literals has a model w.r.t. $\langle \mathcal{O}_{\mathcal{W},\iota}, \mathcal{A} \rangle$ where*

$$\chi_{\mathcal{W},\iota} := \bigwedge_{1 \le i \le k} \chi^{(i)} \wedge \bigwedge_{0 \le i \le n} \chi_{\iota(i)}^{(k+i+1)}, \qquad \chi^{(i)} := \bigwedge_{p_j \in W_i} \varphi_j^{(i)} \wedge \bigwedge_{p_j \in \overline{W_i}} \neg\varphi_j^{(i)},$$

$$\mathcal{O}_{\mathcal{W},\iota} := \{\alpha^{(i)} \mid \alpha \in \mathcal{O},\ 1 \le i \le k + n + 1\}, \qquad \mathcal{A} := \bigcup_{\substack{0 \le i \le n, \\ \alpha \in \mathcal{A}_i}} \{\alpha^{(k+i+1)}\}.$$ □

The original proof again targets TCQs but similarly applies in our more general setting.

To sum up, the TQ satisfiability problem (w.r.t. a TKB) is reduced to two separate ones in [BGL12; BBL15b]: one in LTL and one or several "atemporal" ones in $\mathcal{DL}$—assuming that the set $\mathcal{W}$ and mapping ι, which link the two parts, are given. The intuition is that $\mathcal{W}$ contains exactly the worlds occurring in the LTL model $\mathfrak{W}$ and that ι designates the worlds of the first $n+1$ time points. Each world induces a set of $\mathcal{QL}$ query literals. The DL part then checks whether these sets are consistent (w.r.t. certain classical KBs), to ensure that a DL-LTL structure satisfying the $\mathcal{QL}$ queries (and the TKB) according to $\mathfrak{W}$ can indeed exist.

3.3 Problem Analysis and Technical Contributions

In this work, we focus on the temporal query languages and reasoning problems introduced in this chapter. We consider various different settings:

- We investigate the combined complexity of satisfiability in $\mathcal{DL}$-LTL, both combined and data complexity of TCQ entailment, and rewritability of temporal query answering for various query languages $\mathcal{QL}$.
- In all three parts, we consider $\mathcal{EL}$ and different *DL-Lite* fragments as instantiations of $\mathcal{DL}$. In the last chapter, we also regard various other lightweight logics.
- We consider different settings w.r.t. the rigid symbols and distinguish if no rigid names, only rigid concept names, or both rigid concept and role names are allowed.

For solving satisfiability and entailment, we generally apply the approach presented in the previous section (see Lemma 3.13). That is, for deciding the $\mathcal{DL}$-LTL and the TCQ satisfiability problem, we focus on the following three problems of

(i) obtaining $\mathcal{W}$ and ι,

(ii) solving the LTL satisfiability test (t-satisfiability), and

(iii) solving the DL satisfiability test(s) (r-satisfiability).

Regarding the choice of methods to obtain $\mathcal{W}$ and ι for (i), we have that each can be obtained by enumeration, guessing, or direct construction, depending on the complexity class we target. The size of $\mathcal{W}$ is exponential in that of the considered query Φ, and the size of the mapping ι is linear in n, but there are exponentially many possible such mappings. Note that, in the special case where we have the trivial ABox sequence $(\emptyset)_{0\leq i\leq n}$, a mapping ι can always be obtained easily if there is a set $\mathcal{W}$ that satisfies all but the last conditions of t-satisfiability and r-satisfiability, respectively. To see this, observe that $\mathcal{W}$ must not be empty, and that the last condition in Definition 3.12 requires the world $W_{\iota(0)}$, an element of $\mathcal{W}$, to be consistent w.r.t. the knowledge base $\langle\mathcal{O},\emptyset\rangle$, which is true for every world in $\mathcal{W}$ if $\mathcal{W}$ satisfies the other conditions in that definition. Given that the other conditions are met, the last condition in Definition 3.11 can thus easily be satisfied by defining $\iota\colon [0,0]\to[1,k]$ such that it maps 0 to the index of the world from $\mathcal{W}$ that exists because of the first condition. We state that as a fact for future reference.

Fact 3.15 *For a TQ Φ, TKB $\langle \mathcal{O}, (\emptyset)_{0 \leq i \leq n} \rangle$, and set $\mathcal{W}$ that satisfies all but the last conditions in Definitions 3.12 and 3.11, there exists a mapping $\iota \colon [0, n] \to [1, k]$ such that the remaining conditions are also satisfied.*

Problem (ii) is generally independent of the query language $\mathcal{QL}$ and DL $\mathcal{DL}$. Moreover, it can be solved in exponential time w.r.t. combined complexity and in polynomial time w.r.t. data complexity, by using a reduction to the emptiness of a Büchi automaton [BBL15b, Lem. 4.12].

Regarding Problem (iii), we have two characterizations of r-satisfiability, Definition 3.12 and Lemma 3.14. The critical point with the former is the requirement that the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ share a common domain, because the satisfiability tests for different interpretations cannot be done independently of each other, if rigid symbols are considered. Observe that $\mathcal{J}_1, \dots, \mathcal{J}_k$ characterize the satisfiability of the conjunctions χ_i w.r.t. $\langle \mathcal{O}, \emptyset \rangle$, $i \in [1, k]$, respectively. Moreover, the functions of the common domain are mainly two, which gets more evident if the additionally required respect of rigid symbols is considered:

(F1) Synchronize the interpretation of rigid symbols regarding the named individuals.

(F2) Guarantee that the satisfiability of the conjunctions χ_i, $i \in [1, k]$, which is represented by the respective interpretations $\mathcal{J}_i$, is not contradicted by the interpretation of the rigid names in the other interpretations.

Problem (iii) thus depends on $\mathcal{QL}$ and $\mathcal{DL}$, and especially on which symbols are allowed to be rigid. This can be similarly seen by considering Lemma 3.14. If rigid names are not considered, then the k conjunctions $\chi^{(i)}$ with $i \in [1, k]$ in the conjunction $\chi_{\mathcal{W},\iota}$ do not share any symbols. That is, their satisfiability can be tested independently of each other. This is important considering the fact that the size of $\chi_{\mathcal{W},\iota}$ depends on the exponential number k.

Lemma 3.16 ([BGL12, Lem. 5.1]) *Let $\chi_1, \dots, \chi_\ell$ be conjunctions of $\mathcal{QL}$ query literals, $\mathcal{O}_1, \dots, \mathcal{O}_\ell$ be ontologies, and $\mathcal{A}_1, \dots, \mathcal{A}_\ell$ be ABoxes such that elements with different index $i \in [1, \ell]$ do not share concept and role names. Then the conjunction $\chi_1 \wedge \dots \wedge \chi_\ell$ is satisfiable w.r.t. the KB $\langle \bigcup_{1 \leq i \leq \ell} \mathcal{O}_i, \bigcup_{1 \leq i \leq \ell} \mathcal{A}_i \rangle$ iff, for each $i \in [1, \ell]$, χ_i is satisfiable w.r.t. $\langle \mathcal{O}_i, \mathcal{A}_i \rangle$.* □

Note that [BGL12, Lem. 5.1] refers to the language $\mathcal{QL}$ of $\mathcal{ALC}$ axioms, but the result obviously also holds in our more abstract setting.

Rigid names, on the other hand, generally require the consideration of the whole conjunction. We thus get only rather high upper bounds from Lemma 3.14.

Lemma 3.17 *If the satisfiability problem for conjunctions of $\mathcal{QL}$ query literals w.r.t. a $\mathcal{DL}$ KB is contained in* NP*, then the satisfiability problem for temporal $\mathcal{QL}$ queries w.r.t. a $\mathcal{DL}$ TKB is contained in* NExpTime *(*NP*), and the entailment problem is contained in* co-NExpTime *(*co-NP*), w.r.t. combined (data) complexity.*

Proof. We focus on the satisfiability problem and regard the conditions in Lemma 3.13. Given the above observations, (i) both the set $\mathcal{W} \subseteq 2^{\{p_1, \dots, p_m\}}$ of worlds and the mapping $\iota \colon [0, n] \to [1, k]$ can be nondeterministically guessed in exponential (polynomial)

time, and (ii) t-satisfiability can be decided in exponential (polynomial) time in the input (data). Regarding (iii) r-satisfiability, we have that $\chi_{\mathcal{W},\iota}$ is of size exponential in $|\Phi|$ because of the first k conjuncts. The size of each of the other n conjuncts is $\log(n)$ times (i.e., for the representation of the index) the sum of a number linear in $|\Phi|$, for the conjuncts $\chi^{(k+i+1)}$, $i \in [0, n]$. The size of $\mathcal{O}_{\mathcal{W},\iota}$ is exponential in $|\Phi|$, and linear in $|\mathcal{O}|$ and n. The size of $\mathcal{A}$ is $\log(n)$ times the sum of all $|\mathcal{A}_i|$, $i \in [1, n]$, and thus polynomial in the data. Given the assumption (and the fact that $|\Phi|$ and $|\mathcal{O}|$ are constants), the satisfiability of the conjunction in Lemma 3.14 can be tested in nondeterministic exponential (polynomial) time. TQ satisfiability is thus contained in NExpTime (NP), and entailment in co-NExpTime (co-NP). □

The above observations show that, in many cases, we cannot directly follow the approach of [BGL12; BBL15b] because we target considerably lower complexity results. While the size of ι is only an obstacle for designing algorithms of sublinear complexity, the exponential size of $\mathcal{W}$ makes it impossible to guess (and store) this set by using only a polynomial amount of space. And known results only allow to solve Problems (ii) and (iii) in exponential time. The trivial approach of Lemma 3.13 does hence neither provide an upper bound of PSpace w.r.t. combined complexity, the lower bound we have from LTL satisfiability, nor tractable data complexity, if rigid symbols are considered. Yet, for $\mathcal{EL}$ and the Horn fragments of *DL-Lite*, we usually have such reasoning complexities. Indeed, we obtain such PSpace and tractability results, even in some settings with rigid symbols. More specifically, our contributions are as follows:

- We propose a new, general procedure for solving the TQ satisfiability problem in polynomial space w.r.t. combined complexity based on adapting the algorithm originally proposed for solving the LTL satisfiability problem, which is specified in Algorithm 2.1. Recall that the latter algorithm iteratively constructs a model for a given LTL formula φ: it considers a sequence of exponentially many time points i, iteratively regards each of them, guesses a world w_i, and describes polynomial-space tests ensuring the adequacy of w_i (i.e., regarding the worlds guessed in previous steps) and relying on a polynomial amount of information that is kept and updated during the iteration. We extend that algorithm to iteratively construct a model for a given TQ w.r.t. a TKB. That is, Problems (i) and (ii) are solved in general for Φ^{pa}, independently of $\mathcal{QL}$ and $\mathcal{DL}$; specifically, we use the world w_i guessed in the original algorithm[7] to determine the element W of $\mathcal{W}$ that represents the $\mathcal{QL}$ queries satisfied at i, and ι.[8] But our procedure still has to be tailored to the specific problem under consideration (e.g., TCQ entailment in $\mathcal{EL}$) regarding Problem (iii). Our approach is sketched in Algorithm 3.1. The highlighted parts represent the critical extensions of Algorithm 2.1 (i.e., extensions that may influence correctness or complexity); the functions which target the r-satisfiability testing are specified later in this work for each considered problem individually.

[7] Recall that w_i is determined by $\mathcal{F}_{pres}$.

[8] To determine ι in this way, we have to ensure that the first n time points are considered. This can be done by guessing the start s of the period such that it is large enough. We also have to require that $\Phi^{\text{pa}} \in \mathcal{F}_{pres}$ when $i = n$ instead of when $i = 0$.

Algorithm 3.1: Procedure for Deciding TQ Satisfiability

Input: TQ Φ, TKB $\mathcal{K}$ $\langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$
Output: *true* if Φ is satisfiable w.r.t. $\mathcal{K}$, otherwise *false*

1 $d :=$ `GUESSDATA`$(\Phi, \mathcal{K})$
2 $i := 0$
3 $s :=$ Guess a number $\leq 2^{|\Phi^{\mathrm{pa}}|}$, > 0, and such that $s \geq n$
4 $p :=$ Guess a number $\leq 4^{|\Phi^{\mathrm{pa}}|}$
5 $\mathcal{F}_{next} := \emptyset$, $\mathcal{F}_s := \emptyset$, $\mathcal{F}_{\mathcal{U}} := \emptyset$
6 $\mathcal{F}_{pres} :=$ Guess a subset of Clo$(\{\Phi^{\mathrm{pa}}\})$
7 **if** ***not*** `CONSISTENT`$(\mathcal{F}_{pres})$ **or** ***not*** `INITIAL`$(\mathcal{F}_{pres})$ **then**
8 return *false*
9 **while** $i \leq s + p$ **do**
10 Update $\mathcal{F}_{pres}$ and $\mathcal{F}_{next}$, and proceed as in Algorithm 2.1 for input Φ^{pa}, Lines 9 to 17
11 **if** $i = n$ ***and*** $\Phi^{\mathrm{pa}} \notin \mathcal{F}_{pres}$ **then** return *false*
12 $W := \mathcal{F}_{pres} \cap \{p_1, \ldots, p_m\}$
13 **if** $i > n$ **then** $\mathcal{A}_i := \emptyset$
14 **if** ***not*** `TESTRSAT`$(\Phi, \mathcal{O}, \mathcal{A}_i, d, i, s, p, W)$ **then**
15 return *false*
16 $i := i + 1$
17 Continue as in Algorithm 2.1, Line 19

In particular, this r-satisfiability testing has to be done as follows for obtaining a concrete, polynomial-space algorithm:

- It can rely on a *polynomial amount* of *additional data* d which is guessed in the beginning and kept during the iteration.
- It must be done in tests that require only polynomial space and, during the iteration, may consider only one world W—different from the test described in Lemma 3.14, which considers the exponential set $\mathcal{W}$ as a whole.

Corollaries 4.5, 4.22, 5.16, and 6.19 are obtained based on this new approach.

- The characterizations of r-satisfiability by such a polynomial amount of information and conditions that can be tested by using only polynomial space presents another main contribution of our work. This is because we also target low complexities w.r.t. *rigid symbols*, regarding satisfiability in $\mathcal{EL}$-LTL with global GCIs (Section 4.4.2), and TCQ entailment in $\mathcal{EL}$ (Section 5.1) and Horn fragments of *DL-Lite* (Section 6.1). The common idea of our characterizations is to adapt Definition 3.12: we look for similar interpretations but consider their domains to be

disjoint w.r.t. the unnamed individuals; and we use the additional data and test conditions to achieve the effects of the shared domain (Functions (F1) and (F2)).

- For obtaining tractable data complexity regarding TCQ entailment in $\mathcal{EL}$ and Horn fragments of *DL-Lite*, we also integrate the tests described in Lemma 3.13 into the construction of $\mathcal{W}$ and ι. The main achievement represents the algorithm we propose for *DL-Lite*; there, we include rigid symbols and target the sublinear ALogTime complexity. These results are stated in Theorems 5.19 and 6.37.

- The remaining containment results are rather straightforward consequences of Lemma 3.13 and the above observation that, if rigid symbols are disregarded, the r-satisfiability test can be split into exponentially many satisfiability tests, each regarding only polynomially large conjunction of $\mathcal{QL}$ query literals: Corollaries 4.6, 5.17, and 5.20, and Theorems 7.6 and 7.8.

- Another major contribution are the various hardness results, where we apply the few means these lightweight DLs offer for showing hardness for co-NP, NExpTime, co-NExpTime, and 2-ExpTime in the presence of rigid symbols: Theorems 4.8, 5.18, 5.21, 7.7, and 7.9.

- For TCQ entailment in *DL-Lite* logics between $\textit{DL-Lite}_{krom}$ and $\textit{DL-Lite}_{bool}^{\mathcal{H}}$, we establish close relationships to TCQ entailment in more expressive DLs: Lemmas 7.2 and 7.11.

- In Chapter 8, we lastly obtain a generic rewritability result for answering TQs without negation, which applies to various concrete query languages $\mathcal{QL}$ and lightweight logics—as ontology languages—that have been proposed in the literature.

Note that the following chapters refer to the preliminaries but are self-contained otherwise. This is why explanations in different chapters may overlap due to the above described similarity of the approaches.

We below provide an overview of the assumptions we make throughout Chapters 4 to 7—recall that all of them are without loss of generality:

- Individual names are always rigid.

- Concept and role names occurring in an ABox of a (T)KB also occur in its ontology.

- Individual names occurring in a TCQ also occur in the ABoxes of the (T)KB regarded.

- CQs are connected.

- CQs contained in a TCQ are Boolean and do not share variables; this also applies to multiple occurrences of one CQ.

Especially note that, if we refer to settings without rigid symbols or with only rigid concept names, this does not include the individual names.

Lastly, observe that a rigid concept name A can easily be represented using a flexible concept name A' and a rigid role name R_A: by replacing every occurrence of A by A' and considering the global CIs $A' \sqsubseteq \exists R_A.\top$ and $\exists R_A.\top \sqsubseteq A'$. That is, in $\mathcal{DL}$-LTL, the latter are prefixed by $\Box_F$ and added as conjuncts to the formula and, regarding TCQs, they are added to the ontology. In the context of Chapters 4 to 7, where all the considered DLs allow to express such CIs and global CIs can be expressed in both settings we investigate, DL-LTL satisfiability and TCQ entailment w.r.t. a TKB (with non-empty ontology), the following fact thus holds.

Fact 3.18 *TQ satisfiability in the presence of rigid concept names can be linearly reduced to TQ satisfiability in the presence of rigid role names.*

That is, hardness results for the case $\mathsf{N_{RC}} \neq \emptyset$ also apply to the case where $\mathsf{N_{RR}} \neq \emptyset$, and we can disregard the case where $\mathsf{N_{RC}} = \emptyset$ but $\mathsf{N_{RR}} \neq \emptyset$. Note that we in the following proceed in this way, without explicitly referring to Fact 3.18.

3.4 Related Work

In this section, we review related work on temporal ontology-based data access that is closest to our work, where the ontologies are written in DLs. In general, there are various ways to represent time in DL modeling; for example, by considering time points as concrete datatypes [BH91; Lut01] or formalisms inspired by action logics [AF98; Har+13]. Good overviews of different such approaches are provided in [AF00; AF05].

We focus on *temporal description logics* that are combinations of standard temporal logics with DLs and focus on two-dimensional semantics, which is nowadays the general approach and common understanding of the notion (see Definition 3.1 and the part below); though, there is no formal definition.[9] Such combinations still offer a wealth of degrees of freedom:

- base DL;
- temporal logic (e.g., LTL or CTL, which is shortly covered later in this section);
- application of temporal logic operators: for temporalizing concepts or roles (i.e., if the operators are allowed within concept or role expressions); axioms, such as in $\mathcal{DL}$-LTL; or queries, such as in TQs;
- semantics (e.g., standard first-order semantics or *epistemic semantics*, which separate the temporal and DL dimension (see [Cal+07a], for example)).

Note that the choice of temporal logic fixes the notion of time considered (e.g., if it is based on points or intervals) and that the semantics also specifies details such as if the notion of time is based on the naturals, integers, or reals; and bounded to the past or future. However, we do not further consider specifics such as the latter since their relevance is minor w.r.t. the results we review. Moreover, the approaches in the literature differ w.r.t. the problem and kind of complexity they focus on.

[9] Note that temporal description logics represent special kinds of extensions of DLs with modal logics [Kur+03].

Earlier works investigate temporal versions of standard reasoning problems w.r.t. combined complexity and target applications such as terminologies with temporal aspects or temporal conceptual modeling. In contrast, most recent investigations focus on query answering with the goal of retrieving (temporal) data and also consider data complexity. In the following, we cover both directions.

Regarding the problems studied, we apply the terminology of the atemporal setting (e.g., the satisfiability problem of a concept w.r.t. a TKB is defined in correspondence with Definition 2.5). Moreover, we also consider TKBs where the concepts and roles in the ontology are temporalized, thus extending the TKBs from Definition 3.2. To avoid confusion, we may further slightly adapt the original names of the logics since there is no common naming scheme. Generally, we prefix DLs with the letter T if the logic allows for temporalizing concepts—we mention the temporal logic used—and add a corresponding superscript if only a subset of the temporal logic operators is allowed; the suffix -LTL denotes the fact that axioms may be temporalized with LTL operators.

3.4.1 Reasoning with Temporalized Concepts and Axioms

The first combination of a DL and a point-based temporal logic, based on a two-dimensional semantics, has been proposed in [Sch93]. Subsequent studies then have focused on classifying different combinations of LTL and (extensions of) $\mathcal{ALC}$ regarding expressivity and complexity, with results mostly in the range of ExpSpace, if rigid roles are disregarded. Since a detailed discussion of these results is beyond the scope of this work, we refer to [Kur+03; AF05] for details; the Description Logic Handbook [Baa+07] also provides a summary of main results. An important outcome of that research is however the observation that rigid roles or several other constraints on roles (e.g., temporal operators) generally lead to undecidability. Because decidability represents one major feature of DLs, research since then has been dedicated to the study of decidable temporal DLs, which we review in the following. Decidable temporal DLs are generally obtained by either restricting the expressivity of the temporal features (e.g., restricting the operators or rigid symbols allowed) or the DL (e.g., by focusing on lightweight DLs). Most recent studies often consider restrictions on both sides since they target tractable logics.

[LWZ08] show that, without rigid symbols, neither temporalizing $\mathcal{ALC}$ concepts nor GCIs leads to an increase in complexity over the ExpTime complexity of satisfiability in $\mathcal{ALC}$. Note however that rigid concept names A with the former kind of temporalization can be modeled with GCIs of the form $A \sqsubseteq \Box_F A$. While the temporal and description logic part thus cannot interact critically in these formalisms, the combination of the two allows intricate interactions and leads to ExpSpace-completeness. In a nutshell, this is the case because the LTL part, which generally leads to a model (i.e., a DL-LTL structure) that is periodic with a period of exponential length, here does not have this property any more; formulas as the following example from [LWZ08, p. 8] have only models whose interpretations all must be distinct:

$$\Box_F(\neg(\top \sqsubseteq \neg A) \wedge (A \sqsubseteq \bigcirc_F \Box_F \neg A)).$$

Temporalized DL axioms (see Section 3.1) have originally also been proposed regarding $\mathcal{ALC}$ [BGL12]. The satisfiability problem in $\mathcal{ALC}$-LTL similarly has the same

complexity as in the atemporal case, but rigid concepts and roles lead to NExpTime and 2-ExpTime-completeness, respectively, and thus to a considerable increase in complexity. In the same paper, it is however also shown that this increase can be overcome, at least for rigid concepts, if GCIs are only allowed to occur globally in the formulas. [Lip14] extends the former results to $\mathcal{SHOQ}$-LTL, which implies that they hold for all such temporal DLs based on a DL between $\mathcal{ALC}$ and $\mathcal{SHOQ}$.

Tractable DLs in combination with LTL are investigated in [Art+07; Art+14b]. [Art+14b] investigate the KB consistency problem for several temporal extensions of different *DL-Lite* logics that allow for temporalizing concepts with subsets of LTL operators, including rigid roles. The positive results include containment in NLogSpace or P and are obtained by reduction to fragments of propositional temporal logic, but they only hold for formalisms strongly constrained on the temporal side. In most cases, more variety in that direction leads to NP-completeness and, if more expressive role expressions are allowed (e.g., arbitrary RIs), even to undecidability. Regarding both *DL-Lite* and $\mathcal{EL}$, [Art+07] integrate temporalized concepts, axioms, and rigid roles. They show that the satisfiability of formulas in this setting for the resulting logics *TDL-Lite*$_{bool}$-LTL and *TDL-Lite*$_{horn}^{\bigcirc_F}$-LTL is ExpSpace-complete, whereas it is PSpace-complete in *TDL-Lite*$_{krom}^{\bigcirc_F}$-LTL; *TDL-Lite*$_{bool}$ denotes the temporal extension of *DL-Lite*$_{bool}$ where the concepts in CIs may be formed by applying full LTL to basic concepts while, in the latter two formalisms, only $\bigcirc_F$ may be applied. The logics thus allow to describe necessary future consequences in the ontology; for example, the fact that an application that is out of user focus during two consecutive observations should be in the energy saver mode at the next observation moment, can be expressed as follows:

$$\texttt{Application} \sqcap \texttt{OutOfFocus} \sqcap \bigcirc_F \texttt{OutOfFocus} \sqsubseteq \bigcirc_F \bigcirc_F \texttt{EnergySaverMode}.$$

Note that, in all these formalisms, rigid concept names A can be modeled by adding conjuncts of the form $\Box_F(A \sqsubseteq \bigcirc_F A)$ to the formula. The results of [Art+07] are interesting because, in the atemporal case, complexity results for *DL-Lite*$_{krom}$ are usually worse than corresponding ones for *DL-Lite*$_{horn}$ [Art+09]. In a nutshell, *TDL-Lite*$_{horn}^{\bigcirc_F}$ CIs are critical because they allow to encode a standard counter—over time—that discerns exponentially many time points per individual (e.g., similar to the counter in the proof of Theorem 7.9); in addition, the outer LTL allows to specify conditions that hold for all individuals *per time point*; and additional restrictions on selected individuals, which are based on former time points, may constrain the latter conditions further. Note that this interaction does not depend on rigid roles. The other results are again based on translations into fragments of first-order temporal logic. For $\mathcal{TEL}^{\Diamond_F}$, a logic allowing for temporalized concepts that are standard $\mathcal{EL}$ concepts augmented with the operator $\Diamond_F$, subsumption is shown to be undecidable in the presence of rigid roles; the point is that $\Diamond_F$ in concept expressions in combination with the $\mathcal{EL}$ features allows to model disjunction and hence to reduce to an undecidable temporal version of $\mathcal{ALC}$. For that reason, the complexity of the $\mathcal{EL}$-LTL satisfiability problem was an interesting open question since the logic represents another temporalization of the popular DL $\mathcal{EL}$ and there was no hardness result apart from the PSpace complexity of LTL. We address $\mathcal{EL}$-LTL in [BT15b], where we investigate complexity w.r.t. different selections of rigid symbols and show that $\mathcal{EL}$-LTL can be allocated in between *DL-Lite*-LTL and $\mathcal{ALC}$-

LTL/$\mathcal{SHOQ}$-LTL. These results together with new ones on *DL-Lite*-LTL and other such temporal lightweight DLs are detailed in Chapter 4.

There are some recent works that temporalize concepts with temporal logic operators different from the LTL ones. [GJL12; GJS14; GJS15] consider *computation tree logic* (CTL) [CE81] for temporalizing concepts and investigate subsumption. CTL extends LTL in that the time is not considered to be linear regarding future, but branching. Since it allows to quantify over these branches both existentially and universally, it allows to model alternative future behavior. For example, the fact that a dishwasher that is currently filled by someone may be switched on in the next observation moment can be expressed with the following CI:

$$\texttt{Dishwasher} \sqcap \exists \texttt{FilledBy}.\top \sqsubseteq \mathbf{E} \bigcirc_F \texttt{SwitchedOn}$$

where $\mathbf{E}$ denotes the existential path quantifier. The latter expresses that the statement following it represents a possible future, while other future states may exist though. [GJL12] show that, if CTL is applied to $\mathcal{ALC}$, then the complexity does not increase over the atemporal case and propose a tractable temporal DL based on $\mathcal{EL}$, which is however very restricted. Moreover, the results are very negative—nonelementary or undecidable—if rigid roles are allowed, even for $\mathcal{EL}$ [GJS14]. For that reason, follow-up investigations [GJS15] consider restricted ontologies (e.g., acyclic ones) and, indeed, obtain better results. Nevertheless, containment in PSpace, for instance, is only achieved by either strongly restricting the allowed temporal operators or disregarding the ontology entirely.

A new direction of research has recently been pursued in [GJO16; Baa+17], investigating *metric* versions of LTL for temporalizing $\mathcal{ALC}$ concepts (and GCIs) and regarding satisfiability (of the resulting ontological formulas). One such logic allows, for example, to annotate $\mathcal{U}$ with an interval based on concrete values, representing time points. Yet, the complexities are rather high, often in the range of ExpSpace and 2-ExpSpace. While the papers present important foundational work on metric temporal DLs, the authors themselves state that the need for similar studies regarding lightweight DLs, query answering, and data complexity to analyze the applicability for practical temporal OBDA.

Observe that the above described results also apply to instance query answering (i.e., the problem of instance checking), as it is the case in the atemporal setting. The works do however not investigate data complexity, which is relevant in practice. The more recent studies described next address this issue.

3.4.2 Querying for Temporal Data

Query answering with the goal of retrieving data is in focus of recent research in DLs, in general, and this is reflected in the latest explorations on reasoning about temporal knowledge. The different works are primarily discerned regarding the ontology considered, depending on whether it is also temporal or written in a classical DL. The former approaches offer more expressivity but, on the other hand, tend to lead to higher reasoning complexities. For this reason, they are usually studied w.r.t. lightweight DLs.

[Art+15a] investigate different temporal extensions of the *DL-Lite* logics described in Section 2.1.[10] The goal of the work are first-order rewritability results regarding the temporal query answering problem (see Definition 2.22). The queries are arbitrary combinations of temporalized concept and role atoms using the operators of first-order temporal logic and thus very expressive, which is why epistemic semantics are employed (i.e., it is well known that queries with negation are not tractable, even in the atemporal setting [Gut+15]). However, the investigations then mainly consider a very restricted setting where roles are disregarded and the ABoxes contain a single individual name only (the name itself may occur multiple times in an ABox). Though, these results can, amongst others, be applied to show first-order rewritability of instance query answering in $\mathit{TDL\text{-}Lite}_{core}^{\bigcirc_F}$ and $\mathit{TDL\text{-}Lite}_{core}^{\Box_F}$. In particular, [Art+15a] define specific temporal canonical interpretations for TKBs in these logics, which can be used to construct the rewritings.

Also for TKBs where the ontology is written in $\mathcal{TEL}^{\bigcirc *}$ (i.e., the logic allows for both $\bigcirc_F$ and $\bigcirc_P$), there are canonical models, which can be used to show that the satisfiability problem is tractable w.r.t. data complexity if rigid roles are disallowed [GJK16]. The latter paper also identifies a certain *periodicity* of the ontology to ensure decidability and proposes several acyclicity notions for ontologies in temporal $\mathcal{EL}$ that yield tractable combined and data complexity.

Research in the second direction—temporal query answering regarding classical ontologies—has been first considered in a general way in [GK12] regarding expressive operators on both the temporal and the DL side (i.e., first-order formulas). Temporal conjunctive queries have first been studied in [BBL13], focusing on the complexity of the entailment problem. That paper and follow-up investigations [BBL15b; BBL15a] focus on $\mathcal{ALC}$ and extensions such as $\mathcal{SHOQ}$—under open world semantics—, which is reflected in the rather high complexity results. Yet, for many of the considered DLs the data complexity is in co-NP, as in the atemporal case, even in the presence of rigid concept names. Observe that the exact complexity w.r.t. rigid roles is still open, but containment in ExpTime is known. The combined complexity is generally as in the atemporal case if rigid symbols are not considered, but w.r.t. rigid symbols increases up to 2-ExpTime.

Together with [Kla13; KM14b] we regard TCQ answering and entailment w.r.t. the most prominent lightweight description logics [BLT15; BT15b; BT15a]. Our complexity results on TCQ entailment [BT15b; BT15a] are detailed in Chapters 5, 6, and 7 regarding $\mathcal{EL}$, $\mathit{DL\text{-}Lite}_{core}$ to $\mathit{DL\text{-}Lite}_{horn}^{\mathcal{H}}$, and extensions of $\mathit{DL\text{-}Lite}_{horn}^{\mathcal{H}}$, respectively; note that especially w.r.t. combined complexity, the latter logics have however turned out to be not very "lightweight" any more. In Chapter 8, we describe the initial parts of the work in [BLT15], by generalizing TCQs and proposing a practical approach for temporal query answering; for details about algorithms, especially including rigid concept names, we refer to the paper. In a nutshell, the feasibility of rewriting is achieved by dropping the negation operator. [Kla13; KM14b] also describe an approach for implementation w.r.t. $\mathit{DL\text{-}Lite}_{core}$, but achieve the first-order rewritability by considering epistemic semantics. Note that there are also some works on non-standard reasoning problems re-

[10] Note that [Art+15a] actually study fragments that are slightly more expressive than those described in Section 2.1 (e.g., they consider more expressive RIs that may even model rigid roles).

garding TCQs that are different from query answering and entailment [KM13; KM14a]. There, the focus is on ABox abduction, which roughly is the question of how the given ABox sequence can be extended so that a specific TCQ is entailed.

Observe that, although the above described queries considered in [Art+15a] are very expressive, they do not subsume TCQs. TCQs allow to existentially quantify variables occurring in conjunctions of atoms, whereas the other queries can only express tree-shaped CQs, via concept expressions.

A similar observation can be made regarding the logics where the axioms are temporalized. While formulas containing only temporalized assertions can be directly regarded as TCQs, the latter cannot express local CIs. On the other hand, CQs that are not tree-shaped (e.g., see the introductory example CQ (1.1)) cannot be modeled through concepts in logics such as $\mathcal{ALC}$-LTL.

There are also some recent proposals of interval temporal description logics that allow for tractable query answering [Art+14a; Art+15b]. Specifically, they combine fragments of the Halpern-Shoham interval logic [HS91a] with *DL-Lite*. Note that this setting is rather different from the one we consider. Nevertheless, these approaches follow an interesting direction of research, given that, for example, the possibility to associate data with intervals has been recently incorporated into the SQL standard [KM12].

4 LTL over Lightweight Description Logic Axioms

In this chapter, we focus on an abstract lightweight DL $\mathcal{DL}$, regard a formula Φ in $\mathcal{DL}$-LTL, and investigate the combined complexity of the satisfiability problem in various settings. $\mathcal{DL}$-LTL formulas allow to temporalize ontological axioms and hence, for example, to express rules about temporal knowledge. The following $\mathcal{EL}$-LTL formula describes that it always must hold that, if all occupants in a room are sleeping at two consecutive observation moments, then all lights and screens are switched off at the second time of observation:

$$\Box_F\Big((\texttt{RoomOccupant} \sqsubseteq \texttt{Sleeping}) \wedge \bigcirc_F(\texttt{RoomOccupant} \sqsubseteq \texttt{Sleeping}) \rightarrow$$
$$\bigcirc_F\,(\texttt{Light} \sqsubseteq \exists\texttt{HasState}.\texttt{SwitchedOff}) \wedge\ \bigcirc_F\,(\texttt{Screen} \sqsubseteq \exists\texttt{HasState}.\texttt{SwitchedOff})\Big).$$

Recall that $\mathcal{EL}$-LTL formulas can also contain assertions, next to CIs; for instance, the fact that Ann is sleeping at two consecutive observation moments should always imply that her devices are not fully powered at the second time of observation:

$$\Box_F\Big(\texttt{Sleeping}(\texttt{ann}) \wedge \bigcirc_F\texttt{Sleeping}(\texttt{ann}) \rightarrow$$
$$\bigcirc_F\,\neg\exists\texttt{HasDevice}.\exists\texttt{HasState}.\texttt{FullPowerMode}(\texttt{ann})\Big).$$

In line with the examples where $\mathcal{DL} = \mathcal{EL}$ we specify $\mathcal{DL}$ by requiring it to satisfy the following properties:

(P1) It allows for conjunction, $\top$, and either $\bot$ or qualified existential restriction.

(P2) Satisfiability of conjunctions of $\mathcal{DL}$ literals can be decided in NP.

Note that *DL-Lite*$_{horn}$ is expressive enough to satisfy the first property. In Section 4.1, we show that *DL-Lite*$^{\mathcal{H}}_{bool}$ is "lightweight" enough to satisfy the second one and that $\mathcal{EL}$ also represents an instance of $\mathcal{DL}$. We then prove in Section 4.2 that satisfiability in $\mathcal{DL}$-LTL is in PSpace and hence not harder than satisfiability in LTL, if rigid symbols are disregarded. The rather negative result that rigid symbols lead to NExpTime-completeness is shown in Section 4.3 based on the NExpTime-hardness proof of $\mathcal{ALC}$-LTL [BGL12, Lem. 6.2]; note that satisfiability in $\mathcal{ALC}$-LTL is however 2-ExpTime-hard if rigid roles are considered. Since complexities of this magnitude nevertheless seem to be out of the scope of usual applications of the lightweight DLs, we in the remainder of the chapter consider restricted formulas, with global CIs. Indeed, for the concrete temporal DLs $\mathcal{EL}$-LTL and *DL-Lite*$^{\mathcal{H}}_{horn}$-LTL, the satisfiability problem then also is in PSpace in the presence of rigid symbols.

Our containment proofs basically follow Lemma 3.13. The goal is thus to find a corresponding set $\mathcal{W}$ of LTL worlds within specific time and space constraints, although the size of this set is exponential in that of Φ. Note that ι is irrelevant by Fact 3.15. Throughout the chapter, we use the notation of Section 3.2.

4.1 Satisfiability of Conjunctions of DL Literals

Since conjunctions of DL literals do not contain temporal operators, we do neither have to focus on a specific time point nor to consider an entire DL-LTL structure. Satisfiability can be decided by regarding a single interpretation (i.e., the remaining parts of the structure are irrelevant and can be chosen arbitrarily, which also means that rigid names have no effects).

For $\mathcal{EL}$, we can therefore use a reduction to the knowledge base consistency problem in the DL $\mathcal{ELO}_\bot$, by creating $\mathcal{EL}$ axioms capturing the semantics of the respective literals.

Lemma 4.1 *Satisfiability of conjunctions of $\mathcal{EL}$ literals can be decided in* P*.*

Proof. The proof is by reduction to the consistency problem in the DL $\mathcal{ELO}_\bot$. That is, for a given conjunction Ψ of $\mathcal{EL}$ literals, we construct an $\mathcal{ELO}_\bot$ knowledge base $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ that is consistent iff Ψ is satisfiable.

The initial sets $\mathcal{O}$ and $\mathcal{A}$ contain all GCIs and, respectively, assertions that occur non-negated in Ψ. For each negative literal $\neg\alpha$ occurring in Φ, $\mathcal{K}$ is then extended according to the kind of $\neg\alpha$ as follows:

- Negated concept assertion $\neg C(a)$: the axioms $\overline{C}(a)$ and $C \sqcap \overline{C} \sqsubseteq \bot$, where $\overline{C}$ is a fresh concept name, are added to $\mathcal{K}$.
- Negated role assertion $\neg R(a, b)$: the axiom $\{a\} \sqcap \exists R.\{b\} \sqsubseteq \bot$ is added to $\mathcal{O}$.
- Negated GCI $\neg(C \sqsubseteq D)$: the axioms $C(a)$, $\overline{D}(a)$, and $D \sqcap \overline{D} \sqsubseteq \bot$, where a is a fresh individual name and $\overline{D}$ is a fresh concept name, are added to $\mathcal{K}$.

It is easy to see that $\mathcal{K}$ is consistent iff Ψ is satisfiable. Since consistency of $\mathcal{ELO}_\bot$ KBs is decidable in polynomial time [BBL05, Thm. 4], the claim thus holds. □

Regarding *DL-Lite*, the proof is even more easy. This is because the complexity of KB consistency is given in [Art+09] for KBs where the ABoxes may contain negated assertions, as in our setting (see Section 2.1.1). A KB is thus obtained from a conjunction of *DL-Lite* literals by replacing each negated CI $\neg(B_1 \sqcap \ldots \sqcap B_m \sqsubseteq B_{m+1} \sqcup \ldots \sqcup B_{m+n})$ similar as above by CIs $B_{m+1} \sqcap \overline{B} \sqsubseteq \bot, \ldots, B_{m+n} \sqcap \overline{B} \sqsubseteq \bot$, and assertions $B_1(a), \ldots, B_m(a)$, $\overline{B}(a)$, where $\overline{B}$ and a are fresh symbols. From the results in [Art+09, Thm. 8.2] we then get the following.

Lemma 4.2 *Satisfiability of conjunctions of DL-Lite literals can be decided*

- *in* P *in DL-Lite*$^{\mathcal{H}}_{horn}$ *and*
- *in* NP *in DL-Lite*$^{\mathcal{H}}_{bool}$*.*

4.2 Without Rigid Names

For the case without rigid names, PSpace-hardness of satisfiability directly follows from PSpace-completeness of the satisfiability problem in propositional LTL. Containment in PSpace is however no direct consequence of known results and shown in this section;

though, note that there is such a result for the temporal DL *TDL-Lite*$^{\bigcirc F}_{krom}$ [Art+07] (see Section 3.4 for a description of that formalism). Specifically, we show it for $\mathcal{DL}$-LTL where $\mathcal{DL}$ is a DL for which the satisfiability problem of conjunctions of $\mathcal{DL}$ literals can be decided in NP. As described in Section 3.3, the direct application of the approach applied in the related work [BGL12; BBL15b], captured in Lemma 3.13, yields only a nondeterministic exponential time procedure, because of the exponential size of the set $\mathcal{W}$.

The idea of our Algorithm 3.1 is to integrate the LTL and the DL satisfiability testing: we test the satisfiability of the LTL formula Φ^{pa} as it is done in Algorithm 2.1 and ensure in parallel that the guessed worlds are such that the respective conjunctions of $\mathcal{DL}$ literals from Definition 3.12 are satisfiable. Since we disregard both rigid names and the TKB, the DL part can be solved in this way, by checking the satisfiability of the k conjunctions in χ_i separately, in exponentially many (possibly nondeterministic) tests, each requiring only polynomial time. We then can consider $\mathcal{W}$ to consist of all worlds guessed, although we never had to construct it in total.

Definition 4.3 Given a $\mathcal{DL}$-LTL formula Φ and disregarding rigid symbols, satisfiability can be decided using Algorithm 3.1 by taking $\langle \emptyset, \emptyset \rangle$ as input TKB, assuming `GUESSDATA` to do nothing, and defining `TESTRSAT`$(\Phi, \mathcal{O}, \mathcal{A}_i, d, i, s, p, W)$ to return *true* iff the below conjunction of $\mathcal{DL}$ axioms is satisfiable:

$$\bigwedge_{p_j \in W} \varphi_j \wedge \bigwedge_{p_j \in \overline{W}} \neg \varphi_j.$$

♢

Observe that, in this special case without rigid symbols and TKB, we do not need additional data and have $n = 0$. The latter means that our additional adaptations regarding n in Algorithm 3.1 do not have any effect w.r.t. Algorithm 2.1.

Lemma 4.4 *The nondeterministic algorithm described in Definition 4.3 decides satisfiability in $\mathcal{DL}$-LTL by using only polynomial space if $\mathsf{N_{RC}} = \emptyset$ and $\mathsf{N_{RR}} = \emptyset$.*

Proof. We define $\mathcal{W}$ as the union of all sets W encountered while running the algorithm.

Then, the correctness follows from Lemma 3.13, Fact 3.15, the correctness of the LTL algorithm (see Lemma 2.21), Definition 3.12 regarding the empty TKB and no rigid symbols (i.e., the k conjunctions χ_i do not share concept or role names), Lemma 3.16, and the fact that our adaptation leads to a negative result iff one of the latter conjunctions is not satisfiable.

The space complexity is a consequence of Lemma 2.21, the fact that the size of the conjunction considered in the r-satisfiability test is linear in that of Φ, and (P2). □

The complexity of satisfiability in $\mathcal{DL}$-LTL is then obtained by combining Lemma 4.4 and the well-known result of Savitch [Sav70, Thm. 1].

Corollary 4.5 *Satisfiability in $\mathcal{DL}$-LTL is in* PSPACE *if $\mathsf{N_{RC}} = \emptyset$ and $\mathsf{N_{RR}} = \emptyset$.*

4.3 With Rigid Names

In this section, we show that rigid concept symbols change the picture. In a nutshell, this is the case because, now, interactions are possible: we cannot only use the LTL

features to discern exponentially many time points and to nondeterministically choose a specific axiom at each of them, but we can also apply the DL part to correspondingly discern exponentially many (rigid) concepts, instantiated by different individuals, and thus "save" the LTL choices invariant to time, at the respective of those individuals—via rigid names. If the axioms chosen are associated with specific individuals in this way, then the DL part may additionally constrain the choice. We prove that it needs a NExpTime Turing machine to decide satisfiability in this setting.

The corresponding containment result directly follows from Lemma 3.17 for $\mathcal{DL}$-LTL because of (P2). Lemmas 4.1 and 4.2 thus let us state the following more specific result.

Corollary 4.6 *Satisfiability in $\mathcal{EL}$-LTL and DL-Lite $^{[\ |\mathcal{H}]}_{[horn|bool]}$-LTL is in* NExpTime*, even if $\mathsf{N_{RR}} \neq \emptyset$.*

NExpTime-hardness can be shown already for the case with rigid concept names, by reducing the 2^{n+1}-bounded domino problem.

Definition 4.7 A *domino system* is a triple $\mathcal{D} = (T, H, V)$, where T is a finite set of *domino types* and $H, V \subseteq T \times T$ are the horizontal and vertical *matching conditions*. Let T be a domino system and $I = t_0, \dots, t_{n-1} \in T^n$ be an *initial condition*, which is a sequence of domino types of length $n > 0$. A mapping $\tau\colon [0, 2^{n+1}-1] \times [0, 2^{n+1}-1] \to T$ is a 2^{n+1}*-bounded solution* of $\mathcal{D}$ respecting the initial condition I iff the following hold for all $x, y < 2^{n+1}$:

- if $\tau(x, y) = t$ and $\tau((x+1) \bmod 2^{n+1}, y) = t'$, then $(t, t') \in H$;
- if $\tau(x, y) = t$ and $\tau(x, (y+1) \bmod 2^{n+1}) = t'$, then $(t, t') \in V$;
- $\tau(i, 0) = t_i$ for $i < n$. ♢

The complexity we target is established in [BGG97, Thm. 6.1.2], where it is shown that there is a domino system $\mathcal{D} = (T, H, V)$ such that, given an initial condition $I = t_0, \dots, t_{n-1} \in T^n$, the problem of deciding if $\mathcal{D}$ has a 2^{n+1}-bounded solution respecting I is NExpTime-hard.

For the reduction, we apply the features outlined above. The exponentially many different time points, each associated with a specific rigid concept and individual instantiating it, represent the positions in the plane of the domino. To tile the plane, we represent the domino types as flexible concepts and require a specific named individual to always satisfy one of them, by nondeterministically choosing the corresponding assertion. The ontology is used to transfer that domino choice to the entire domain (i.e., by using local GCIs of the form $\top \sqsubseteq \dots$) and to save it via a rigid concept at the individual associated to the current time point. The rigid concepts ensure that all positions and the chosen types are instantiated in *every world*, which allows us to enforce the matching conditions.

Theorem 4.8 *Satisfiability in $\mathcal{DL}$-LTL is* NExpTime*-hard if $\mathsf{N_{RC}} \neq \emptyset$, even if $\mathsf{N_{RR}} = \emptyset$.*

Proof. Our proof consists of two steps: we first reduce the 2^{n+1}-bounded domino problem for a given domino $\mathcal{D}$ with initial condition I to checking the satisfiability of a $\mathcal{DL}$-LTL formula $\Phi_{\mathcal{D},I}$ applying $\bot$, and then dispose of the $\bot$ constructor by using qualified

existential restriction. We thus cover both kinds of DLs $\mathcal{DL}$ we consider (see (P1), in the beginning of the chapter). For the reduction, we adapt the NExpTime-hardness proof for $\mathcal{ALC}$-LTL w.r.t. rigid concepts [BGL12, Lem. 6.2]; we point out the differences to that reduction at the end of the proof. We assume a domino system $\mathcal{D} = (T, H, V)$ with initial condition I to be given and construct a $\mathcal{DL}$-LTL formula $\Phi_{\mathcal{D},I}$ such that $\Phi_{\mathcal{D},I}$ is satisfiable iff $\mathcal{D}$ has a 2^{n+1}-bounded solution respecting I. Subsequently, we first list and describe the symbols we use and then specify $\Phi_{\mathcal{D},I}$.

We discern *global*[1] concept names that are flexible and either satisfied by all domain elements or by none of them; *local* concept names are rigid and used to identify specific individuals. Global names are used to transfer values such as domino types from a named individual a to all other elements, and local names are used to save these values at some of these elements. Altogether, we use the following symbols:

- an individual name a;
- flexible (global) concept names $Z_0, \dots, Z_{2n+1}$, $Z_0^h \dots, Z_{2n+1}^h$, $Z_0^v, \dots, Z_{2n+1}^v$, to realize three binary counters modulo 2^{2n+2} (Z, Z^h, and Z^v) over the positions in the plane, by iterating over time;
- rigid (local) concept names $X_0, \dots, X_n$ and $Y_0, \dots, Y_n$, to similarly realize two binary counters modulo 2^{n+1}, where the X-counter describes the horizontal and the Y-counter the vertical position in the plane;
- flexible (global) concept names G_t, G_t^h, G_t^v, and a rigid (local) concept name L_t for all $t \in T$, to describe the tiling;
- rigid (local) concept names $\overline{X}_0, \dots, \overline{X}_n$, $\overline{Y}_0, \dots, \overline{Y}_n$, and flexible (global) concept names $\overline{Z}_0, \dots, \overline{Z}_{2n+1}$, $\overline{Z}_0^h, \dots, \overline{Z}_{2n+1}^h$, and $\overline{Z}_0^v, \dots, \overline{Z}_{2n+1}^v$ representing the complements of the above counters;
- auxiliary flexible concept names N, $E_0^h, \dots, E_{2n+1}^h$, $E_0^v, \dots, E_{2n+1}^v$.

The first $n+1$ bits of the Z, Z^h and Z^v-counters are used to represent the 2^{n+1} x-coordinates, and the second $n+1$ bits are used to represent the y-coordinates of the plane. We count with the Z-counter up to 2^{2n+2} by iterating over time and use the time points to realize all positions $(x, y) \in [0, 2^{n+1} - 1] \times [0, 2^{n+1} - 1]$. Specifically, we enforce a to satisfy the concepts from the corresponding subset of $\{Z_0, \dots, Z_{2n+1}\}$ for every value of the Z-counter.[2] By modeling the concept names Z_i as global, we ensure that the value of the counter is always transferred to all domain elements. For every position represented by the Z-counter, the Z^h and Z^v-counters represent the top and right neighbor position, respectively. Note however that, regarding a specific world, these three counters only allow us to enforce that the position they currently represent—regarding the Z-counter, we call it the *current* position—is instantiated, by all individuals.

To be able to address arbitrary positions independently of the current time point, we use the rigid concept names $X_0, \dots, X_n$ and $Y_0, \dots, Y_n$. In particular, we ensure

[1] Not to be confused with *rigid* or *always* (in time).

[2] For simplicity, we refer to a instead of to "the domain element representing a".

that there is always (at least) one individual whose X and Y-values match the value of the global Z-counter, such that every position $(x, y) \in [0, 2^{n+1} - 1] \times [0, 2^{n+1} - 1]$ is represented by at least one individual in *every world*.

To tile the plane, we ensure that a satisfies exactly one *current* global domino type G_t in every world, which represents the type t chosen for the current position. The local concept L_t is used to represent this choice independently of time, similar to the use of the X/Y-counter w.r.t. the positions represented by the Z-counter. Specifically, we do not only ensure that there is an individual whose X and Y-values match the value of the global Z-counter, but it also must satisfy L_t if G_t is the current global type. In this way, we ensure that all positions and the chosen types are instantiated in every world, which allows us to enforce the matching conditions. For the latter, we explicitly represent the types chosen for the two important neighbor positions as global types G_t^h and G_t^v and ensure that they match the local types of the respective positions.

We now formalize these descriptions and the initial condition and construct the $\mathcal{DL}$-LTL formula $\Phi_{\mathcal{D},I}$ as a conjunction of the $\mathcal{DL}$-LTL formulas listed in the following.

- Every value of the Z-counter is realized in some world by a, which means that a satisfies the concepts from the corresponding subset of $\{Z_0, \dots, Z_{2n+1}\}$:

$$\Box_F \bigwedge_{0 \le i \le 2n+1} \Big(\Big(\bigwedge_{0 \le j < i} Z_j(a) \Big) \leftrightarrow (Z_i(a) \leftrightarrow \bigcirc_F \neg Z_i(a)) \Big).$$

  This formula requires the i-th bit of the Z-counter to be flipped from one world to the next iff all preceding bits are true. Thus, the value of the Z-counter in the next world is equal to the value in the current world incremented by one.

- In every world, the Z^h and Z^v-counter are synchronized with the Z-counter and also realized by a. This is described similar to the Z-counter in two steps. (i) The x-coordinate represented by the Z^h-counter equals the one represented by the Z-counter plus 1:

$$\Box_F \bigwedge_{0 \le i \le n} \Big(\Big(\bigwedge_{0 \le j < i} Z_j(a) \Big) \leftrightarrow (Z_i(a) \leftrightarrow \neg Z_i^h(a)) \Big).$$

  (ii) The y-coordinates represented by the two counters are the same:

$$\Box_F \bigwedge_{n+1 \le i \le 2n+1} (Z_i(a) \leftrightarrow Z_i^h(a)).$$

  The Z^v-counter is modeled correspondingly:

$$\Box_F \bigwedge_{n+1 \le i \le 2n+1} \Big(\Big(\bigwedge_{n+1 \le j < i} Z_j(a) \Big) \leftrightarrow (Z_i(a) \leftrightarrow \neg Z_i^v(a)) \Big),$$

$$\Box_F \bigwedge_{0 \le i \le n} (Z_i(a) \leftrightarrow Z_i^v(a)).$$

- Regarding a, the interpretation of the concept name $\overline{Z}_i$ as the complement of Z_i is enforced as follows; corresponding conjuncts are used to model $\overline{Z}_i^h$ and $\overline{Z}_i^v$, the complements of Z_i^h and Z_i^v, respectively:

$$\Box_F \bigwedge_{0 \le i \le 2n+1} (\overline{Z}_i(a) \leftrightarrow \neg Z_i(a)).$$

- The values of the three global counters Z, Z^h, and Z^v (and their complements) are shared by all individuals in every world. Again, the global formulas for Z^h and Z^v, and also those for the three complements, are analogous to the ones for the Z counter:
$$\Box_F \bigwedge_{0 \leq i \leq 2n+1} ((\top \sqsubseteq Z_i) \vee (Z_i \sqsubseteq \bot)) .$$

- In every world, also one combination of the X and the Y-counter is realized by (at least) one individual. To ensure that all combinations are covered, we consider the one matching the global Z-counter (in this world) and, to make sure that they are realized, we require the concept N to be instantiated in every world. N is then used to model the former:
$$\begin{aligned} \Box_F \Big(\neg(N \sqsubseteq \bot) \wedge \bigwedge_{0 \leq i \leq n} (N \sqcap Z_i \sqsubseteq X_i) \wedge \bigwedge_{n+1 \leq i \leq 2n+1} (N \sqcap Z_i \sqsubseteq Y_{i-(n+1)}) \wedge \\ \bigwedge_{0 \leq i \leq n} (N \sqcap X_i \sqsubseteq Z_i) \wedge \bigwedge_{n+1 \leq i \leq 2n+1} (N \sqcap Y_{i-(n+1)} \sqsubseteq Z_i) \Big). \end{aligned} \tag{4.1}$$
In the same way, we enforce the correct interpretation of the complements of the local counters. That is, we use a formula as the above one where Z_i, X_i, and $Y_{i-(n+1)}$ are replaced by $\overline{Z}_i, \overline{X}_i$, and $\overline{Y}_{i-(n+1)}$, respectively.

Altogether, the above descriptions ensure that every position of the plane is realized in every world by some individual. Note that we need this to enforce the matching conditions given in $\mathcal{D}$ with the help of global CIs, since they address different positions at the same time. Next, we describe the admissible tilings.

- Every world gets exactly one (global) domino type that represents the choice for the current position and is shared by all individuals:
$$\Box_F \Big(\bigvee_{t \in T} \Big((\top \sqsubseteq G_t) \wedge \bigwedge_{t' \in T \setminus \{t\}} (G_{t'} \sqsubseteq \bot) \Big) \Big). \tag{4.2}$$
Again, we similarly consider the global domino types G_t^h and G_t^v representing the choices for the current right and, respectively, top neighbor position (represented by Z^h and Z^v).

- Given the domino types chosen for the neighbor positions, the horizontal and vertical matching condition can be enforced easily:
$$\Box_F \Big(\bigvee_{(t,t') \in H} (G_t(a) \wedge G_{t'}^h(a)) \wedge \bigvee_{(t,t') \in V} (G_t(a) \wedge G_{t'}^v(a)) \Big).$$

- To synchronize the domino types G_t, G_t^h, and G_t^v w.r.t. the different worlds (e.g., type $t \in T$ represented by G_t^h in the current world equals that represented by G_t in the world where the Z-counter is equal to the current Z^h-counter), we use the local (rigid) domino types L_t. First, we ensure that the local type satisfied by

the individual representing the current position equals the current global type G_t with CIs as follows for all $t \in T$:

$$\Box_F \bigwedge_{t \in T} \big((N \sqcap G_t \sqsubseteq L_t) \wedge (N \sqcap L_t \sqsubseteq G_t)\big).$$

Note that, because of the second of the above CIs and the fact that every world has exactly one global domino type G_t, satisfied by all individuals (see (4.2)), every individual satisfying N in some world also has exactly one local domino type: the global type of that world, in which it represents the current position (see (4.1), which also ensures that such an individual exists for every world).

To synchronize the choices of domino types, especially those represented by G_t, G_t^h and G_t^v, we employ the auxiliary concept names E_i^h and E_i^v:

$$\Box_F \bigwedge_{0 \le i \le n} \big((Z_i^h \sqcap X_i \sqsubseteq E_i^h) \wedge (\overline{Z}_i^h \sqcap \overline{X}_i \sqsubseteq E_i^h)\big) \wedge$$
$$\Box_F \bigwedge_{n+1 \le i \le 2n+1} \big((Z_i^h \sqcap Y_{i-(n+1)} \sqsubseteq E_i^h) \wedge (\overline{Z}_i^h \sqcap \overline{Y}_{i-(n+1)} \sqsubseteq E_i^h)\big).$$

The CIs for E_i^v are analogous, using the Z_i^v and $\overline{Z}_i^v$-counters. In this way, the interpretation of $E_0^h \sqcap \cdots \sqcap E_{2n+1}^h$ must include all those domain elements that satisfy the one combination of X and Y-values which matches the current Z^h-counter. This particularly includes the individual instantiating N that is forced to exist in the corresponding world by (4.1). Moreover, it allows us to ensure that the global domino type G_t^h matches the local domino type L_t at all domain elements satisfying $E_0^h \sqcap \cdots \sqcap E_{2n+1}^h$; analogous arguments and CIs apply regarding the vertical direction:

$$\Box_F\big((E_0^h \sqcap \ldots \sqcap E_{2n+1}^h \sqcap G_t^h \sqsubseteq L_t) \wedge (E_0^h \sqcap \ldots \sqcap E_{2n+1}^h \sqcap L_t \sqsubseteq G_t^h)\big).$$

The initial condition $I = t_0, \ldots, t_{n-1}$ is described as follows, for all $i \in [0, n-1]$:

$$\Box_F\big((C_Z^x = i) \sqcap \overline{Z}_{n+1} \sqcap \cdots \sqcap \overline{Z}_{2n+1} \sqsubseteq G_{t_i}\big)$$

where, for all $b_j \in \{0,1\}$ and $j \in [0,n]$,

$$\Big(C_Z^x = \sum_{0 \le j \le n} 2^j * b_j\Big) := \prod_{\substack{0 \le j \le n \\ b_j = 0}} \overline{Z}_j \sqcap \prod_{\substack{0 \le j \le n \\ b_j = 1}} Z_j.$$

This conjunction identifies a particular x-position in the Z-counter. If the y-coordinate represented by the Z-counter is 0 additionally, as enforced above, then the corresponding domino type of the initial condition is enforced. This finishes the definition of the $\mathcal{DL}$-LTL formula $\Phi_{\mathcal{D},I}$, which consists of the conjuncts specified and mentioned above. Observe that the size of $\Phi_{\mathcal{D},I}$ is polynomial in n. Moreover, it can be checked that $\Phi_{\mathcal{D},I}$ is satisfiable w.r.t. $\langle \mathcal{O}_{\mathcal{D},I}, \emptyset \rangle$ iff $\mathcal{D}$ has a 2^{n+1}-bounded solution respecting I.

We lastly describe how the bottom constructor can be eliminated. We apply an idea of [BBL05] (see the proof of Theorem 7), use a fresh *rigid* concept name L and a flexible role name R, and require the following formula to be satisfied:

$$\Phi_L := \neg L(a) \wedge \Box_F(\exists R.L \sqsubseteq L).$$

By replacing the negated CI $\neg(N \sqsubseteq \bot)$ in $\Phi_{\mathcal{D},I}$ by $\top \sqsubseteq \exists R.N$, we ensure that

- in all the exponentially many worlds, a has an R-successor that satisfies N and, because of (4.1), represents the position associated to the corresponding world; and
- a as well as the above mentioned individuals do not satisfy L in any world, since L is rigid.

We hence can use L instead of $\bot$ without changing the semantics everywhere else in the formula $\Phi_{\mathcal{D},I}$. The reason for this is that it suffices to enforce the CIs with $\bot$ on the right-hand side to hold regarding individuals representing the 2^{2n+2} relevant positions, instead of for all domain elements. We denote by $\Phi'_{\mathcal{D},I}$ the formula resulting from these replacements and get that $\Phi'_{\mathcal{D},I} \wedge \Phi_L$ is satisfiable iff $\mathcal{D}$ has a 2^{n+1}-bounded solution respecting I.

Given the above observations, it is easy to see that the polynomial size and correctness are retained. This finishes the reduction, and NExpTime-hardness follows from (P1) and the NExpTime-hardness of the 2^{n+1}-bounded domino problem [BGG97, Thm. 6.1.2]. □

We point out main differences to the proof for $\mathcal{ALC}$-LTL, which similarly reduces the domino problem. The disjunction allowed in $\mathcal{ALC}$ CIs allows to enforce the matching conditions by using only a single global type G_t, while we additionally need G_t^h and G_t^v together with a complex synchronization with the help of the auxiliary concepts. More specifically, $\mathcal{ALC}$-LTL allows to require that the current global type and the local types of the elements representing the important neighbor positions fulfill *some* condition of H and V (for details, we refer to the original proof). In contrast, we have to ensure the matching using the nondeterminism of LTL and hence the $\mathcal{DL}$-LTL formula and the individual a. Another difference is the presence of the concept names explicitly representing the complements of the various counters (e.g., $\overline{X_i}$). In $\mathcal{ALC}$, this can be directly expressed using negation.

4.4 Global GCIs in $\mathcal{EL}$-LTL

In this section, we focus on $\mathcal{EL}$, consider a setting where the GCIs are global, and show that the satisfiability problem then, also in the presence of rigid names, is in PSpace.[3] The restriction to global GCIs has similarly been shown to yield a better complexity for $\mathcal{ALC}$-LTL in the case with rigid concept names, ExpTime instead of the NExpTime-completeness in the case without restrictions [BGL12, Thm. 6.5]. We hence show the impact to be considerably greater if $\mathcal{EL}$ is considered. An $\mathcal{EL}$-LTL formula with global GCIs is generally of the following form

$$\Phi := (\Box \Psi_{\mathcal{O}}) \wedge \Psi \tag{4.3}$$

where $\mathcal{O}$ is an $\mathcal{EL}$ ontology, $\Psi_{\mathcal{O}}$ is the conjunction of the GCIs contained in it, and Ψ is an $\mathcal{EL}$-LTL formula that contains no GCIs and in which all concept assertions are assertions of basic concepts; without loss of generality, we assume the GCIs to be in normal form and all concept and role names occurring in the formula to also occur in $\mathcal{O}$. The generalized TQ answering setting we introduced in Chapter 3 moreover allows us to regard the ontological part of the formula to be the global ontology in the TKB so that we can assume Φ to contain only $\mathcal{EL}$ assertions in the following.

For deciding satisfiability of Φ, we again rely on Lemma 3.13; a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ is thus critical. This set is of exponential size, captures the LTL part of the satisfiability problem in an abstract way, and also describes the axioms to be considered for satisfiability testing in $\mathcal{EL}$. Specifically, each element of $\mathcal{W}$ induces a set of (negated) axioms and, because of the rigid symbols, these sets cannot be considered separately.

In what follows, we first propose *rigid canonical interpretations* of polynomial size to summarize consequences of rigid knowledge in these sets of $\mathcal{EL}$ axioms. Then, we propose a new characterization of r-satisfiability of $\mathcal{W}$ in terms of those interpretations where we consider the elements of $\mathcal{W}$ separately. Hence, it can be applied in our overall approach of Algorithm 3.1 for deciding satisfiability of Φ by using only polynomial space.

4.4.1 Rigid Canonical Interpretations

To explicitly represent knowledge that follows from rigid information in axioms and hence is invariant to time, we define *rigid canonical interpretations*. These interpretations are iteratively constructed based on the rigid knowledge in the finite canonical interpretations (see Definition 2.10) and similarly to the latter. For that, we introduce new unnamed elements that serve as prototypical successors, in the same way the unnamed elements in the canonical interpretations do; in contrast, the latter elements do not play that role any more.

Definition 4.9 The *rigid canonical interpretation* $\mathcal{I}'_{\mathcal{K}}$ for a KB $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ written in $\mathcal{EL}$ is based on the finite canonical interpretation $\mathcal{I}^f_{\mathcal{K}}$ and the set $\Delta_{\mathsf{u}}^{\mathcal{I}'_{\mathcal{K}}}$ of unnamed individuals, defined as follows:

$$\Delta_{\mathsf{u}}^{\mathcal{I}'_{\mathcal{K}}} := \{u'_A \mid A \in \mathsf{N_C}(\mathcal{O}) \cup \{\top\}\}.$$

[3] Note that the notion "global" here has a semantics different from the one of the global concepts considered in the previous section.

- The domain is:

$$\Delta^{\mathcal{I}'_\mathcal{K}} := \Delta^{\mathcal{I}^f_\mathcal{K}} \cup \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}}.$$

- For all $a \in \mathsf{N_I}$ and $A \in \mathsf{N_{RC}}$:

$$\begin{aligned} a^{\mathcal{I}'_\mathcal{K}} &:= a, \\ A^{\mathcal{I}'_\mathcal{K}} &:= A^{\mathcal{I}^f_\mathcal{K}} \cup \{u'_{A_1} \in \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \mid \mathcal{O} \models A_1 \sqsubseteq A\}. \end{aligned}$$

- For all $i \geq 0$, let interpretations $\mathcal{I}'_{\mathcal{K},i}$ be defined as $\mathcal{I}'_\mathcal{K}$ is defined up to this point. Further, they are specified iteratively as follows, for all $A \in \mathsf{N_C} \setminus \mathsf{N_{RC}}$, $R \in \mathsf{N_{RR}}$, and $S \in \mathsf{N_R} \setminus \mathsf{N_{RR}}$:

$$\begin{aligned} A^{\mathcal{I}'_{\mathcal{K},0}} &:= \{u'_{A_1} \in \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \mid \mathcal{O} \models A_1 \sqsubseteq A\}, \\ R^{\mathcal{I}'_{\mathcal{K},0}} &:= R^{\mathcal{I}^f_\mathcal{K}} \cup \{(u'_{A_1}, u'_{A_2}) \in \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \times \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \mid \mathcal{O} \models A_1 \sqsubseteq \exists R.A_2\}, \\ S^{\mathcal{I}'_{\mathcal{K},0}} &:= \{(u'_{A_1}, u'_{A_2}) \in \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \times \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \mid \mathcal{O} \models A_1 \sqsubseteq \exists S.A_2\}, \\ A^{\mathcal{I}'_{\mathcal{K},i+1}} &:= \{e \in B^{\mathcal{I}'_{\mathcal{K},i}} \mid B \in \mathbb{B}(\mathcal{O}), \mathcal{O} \models B \sqsubseteq A\}, \\ R^{\mathcal{I}'_{\mathcal{K},i+1}} &:= R^{\mathcal{I}'_{\mathcal{K},i}} \cup \{(e, u'_{A_1}) \in B^{\mathcal{I}'_{\mathcal{K},i}} \times \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \mid B \in \mathbb{B}(\mathcal{O}),\ \mathcal{O} \models B \sqsubseteq \exists R.A_1\}, \\ S^{\mathcal{I}'_{\mathcal{K},i+1}} &:= S^{\mathcal{I}'_{\mathcal{K},i}} \cup \{(e, u'_{A_1}) \in B^{\mathcal{I}'_{\mathcal{K},i}} \times \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}} \mid B \in \mathbb{B}(\mathcal{O}),\ \mathcal{O} \models B \sqsubseteq \exists S.A_1\}. \end{aligned}$$

- The definition of $\mathcal{I}'_\mathcal{K}$ is finished based on these interpretations:

$$A^{\mathcal{I}'_\mathcal{K}} := \bigcup_{i \geq 0} A^{\mathcal{I}'_{\mathcal{K},i}}, \qquad R^{\mathcal{I}'_\mathcal{K}} := \bigcup_{i \geq 0} R^{\mathcal{I}'_{\mathcal{K},i}}, \qquad S^{\mathcal{I}'_\mathcal{K}} := \bigcup_{i \geq 0} S^{\mathcal{I}'_{\mathcal{K},i}}. \qquad \diamondsuit$$

Since the domain of the rigid canonical interpretation is of polynomial size, it is easy to see that the above iteration converges after polynomially many iterations. Thus the interpretation is constructed in polynomial time.

We now prove two auxiliary lemmas to characterize $\mathcal{I}'_\mathcal{K}$ regarding its domain elements. First, we show that the constituent interpretations of $\mathcal{I}'_\mathcal{K}$ all agree regarding the "new" unnamed domain elements.

Lemma 4.10 *Let $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ be a consistent knowledge base in $\mathcal{EL}$. For all $u'_A \in \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}}$, $B \in \mathbb{B}(\mathcal{O})$, and $i \geq 0$, we have $u'_A \in B^{\mathcal{I}'_{\mathcal{K},i}}$ iff $\mathcal{O} \models A \sqsubseteq B$.*

Proof. The proof is by induction on i and starts with $i = 0$.

- Let $B \in \mathsf{N_C}$. The claim holds directly by the definition of $B^{\mathcal{I}'_{\mathcal{K},0}}$.
- Let $B = \exists R.A_1$. ($\Rightarrow$) Given $u'_A \in (\exists R.A_1)^{\mathcal{I}'_{\mathcal{K},0}}$, the interpretation of roles in $\mathcal{I}'_{\mathcal{K},0}$ and the fact that $u'_A \notin \Delta^{\mathcal{I}^f_\mathcal{K}}$ yield that there must be an element $u'_{A_2} \in \Delta^{\mathcal{I}'_\mathcal{K}}_{\mathsf{u}}$ such that $(u'_A, u'_{A_2}) \in R^{\mathcal{I}'_{\mathcal{K},0}}$ and $u'_{A_2} \in A_1^{\mathcal{I}'_{\mathcal{K},0}}$ and, specifically, that $\mathcal{O} \models A \sqsubseteq \exists R.A_2$. From that, we get $\mathcal{O} \models A_2 \sqsubseteq A_1$ by the definition of $A_1^{\mathcal{I}'_{\mathcal{K},0}}$, and this yields $\mathcal{O} \models A \sqsubseteq \exists R.A_1$. ($\Leftarrow$) If $\mathcal{O} \models A \sqsubseteq \exists R.A_1$, then we have $(u'_A, u'_{A_1}) \in R^{\mathcal{I}'_{\mathcal{K},0}}$ and $u'_{A_1} \in A_1^{\mathcal{I}'_{\mathcal{K},0}}$ by the definition of $\mathcal{I}'_{\mathcal{K},0}$. This means $u'_A \in (\exists R.A_1)^{\mathcal{I}'_{\mathcal{K},0}}$.

In the induction step, we regard $i > 0$.

- Let $B \in \mathsf{N_C}$. $u'_A \in B^{\mathcal{I}'_{\mathcal{K},i}}$ is equivalent to the existence of a concept $C \in \mathbb{B}(\mathcal{O})$ such that $\mathcal{O} \models C \sqsubseteq B$ and $u'_A \in C^{\mathcal{I}'_{\mathcal{K},i-1}}$, which is equivalent to $\mathcal{O} \models A \sqsubseteq C$ by the induction hypothesis. We hence get the equivalence to $\mathcal{O} \models A \sqsubseteq B$.
- Let $B = \exists R.A_1$. $u'_A \in B^{\mathcal{I}'_{\mathcal{K},i}}$ means that an element $u'_{A_2} \in \Delta_{\mathsf{u}}^{\mathcal{I}'_{\mathcal{K}}}$ exists such that (i) $(u'_A, u'_{A_2}) \in R^{\mathcal{I}'_{\mathcal{K},i}}$ and (ii) $u'_{A_2} \in A_1^{\mathcal{I}'_{\mathcal{K},i}}$ hold. By the interpretation of symbols in Definition 4.9, (i) is equivalent to the existence of a $j < i$ and a $C \in \mathbb{B}(\mathcal{O})$ such that $u'_A \in C^{\mathcal{I}'_{\mathcal{K},j}}$ and $\mathcal{O} \models C \sqsubseteq \exists R.A_2$ and (ii) is equivalent to the existence of a concept $D \in \mathbb{B}(\mathcal{O})$ such that $u'_{A_2} \in D^{\mathcal{I}'_{\mathcal{K},i-1}}$ and $\mathcal{O} \models D \sqsubseteq A_1$. The induction hypothesis then, respectively, yields equivalences to $\mathcal{O} \models A \sqsubseteq C$ and $\mathcal{O} \models A_2 \sqsubseteq D$, and we get $\mathcal{O} \models A \sqsubseteq \exists R.A_1$, altogether. □

The second auxiliary lemma describes $\mathcal{I}'_{\mathcal{K}}$ regarding the domain elements of $\mathcal{I}^f_{\mathcal{K}}$.

Lemma 4.11 *Let $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ be a consistent KB in $\mathcal{EL}$, $e \in \Delta^{\mathcal{I}^f_{\mathcal{K}}}$, and $B \in \mathbb{B}(\mathcal{O})$. If $e \in B^{\mathcal{I}'_{\mathcal{K}}}$, then there is a rigid concept C over $\mathcal{O}$ such that $e \in C^{\mathcal{I}^f_{\mathcal{K}}}$ and $\mathcal{O} \models C \sqsubseteq B$.*

Proof. We assume $e \in B^{\mathcal{I}'_{\mathcal{K}}}$. For the case that $B \in \mathsf{N_{RC}}$, Definition 4.9 yields $e \in B^{\mathcal{I}^f_{\mathcal{K}}}$. $\mathcal{O} \models B \sqsubseteq B$ holds trivially. For the remaining cases, the proof is by induction over the construction of $\mathcal{I}'_{\mathcal{K}}$, showing that, for all $i \geq 0$, $e \in B^{\mathcal{I}'_{\mathcal{K},i}}$ implies the existence of a concept C as above. For the base case, let $i = 0$.

- Let $B \in \mathsf{N_C} \setminus \mathsf{N_{RC}}$. This case cannot apply since $B^{\mathcal{I}'_{\mathcal{K},0}} \cap \Delta^{\mathcal{I}^f_{\mathcal{K}}}$ is empty.
- Let $B = \exists R.A$. Given the assumption, there is an element $d \in \Delta^{\mathcal{I}'_{\mathcal{K}}}$ such that $(e, d) \in R^{\mathcal{I}'_{\mathcal{K},0}}$ and $d \in A^{\mathcal{I}'_{\mathcal{K},0}}$. Because of $e \in \Delta^{\mathcal{I}^f_{\mathcal{K}}}$, we must have $R \in \mathsf{N_{RR}}$ and $(e, d) \in R^{\mathcal{I}^f_{\mathcal{K}}}$ by the definition of $\mathcal{I}'_{\mathcal{K},0}$. But then $d \in \Delta^{\mathcal{I}^f_{\mathcal{K}}}$ similarly implies $A \in \mathsf{N_{RC}}$ and $d \in A^{\mathcal{I}^f_{\mathcal{K}}}$. Thus, we can set $C := \exists R.A$.

For the induction step, we assume $e \in B^{\mathcal{I}'_{\mathcal{K},i+1}} \setminus B^{\mathcal{I}'_{\mathcal{K},i}}$.

- Let $B \in \mathsf{N_C} \setminus \mathsf{N_{RC}}$. The assumption and the definition of $B^{\mathcal{I}'_{\mathcal{K},i+1}}$ imply that there is a concept $B_1 \in \mathbb{B}(\mathcal{O})$ such that $e \in B_1^{\mathcal{I}'_{\mathcal{K},i}}$ and $\mathcal{O} \models B_1 \sqsubseteq B$. By the induction hypothesis, there is a rigid concept C over $\mathcal{O}$ such that $e \in C^{\mathcal{I}^f_{\mathcal{K}}}$ and $\mathcal{O} \models C \sqsubseteq B_1$. But then we also have $\mathcal{O} \models D \sqsubseteq B$, as required.
- Let $B = \exists R.A_1$. We can consider an element $d \in \Delta^{\mathcal{I}'_{\mathcal{K}}}$ such that $(e, d) \in R^{\mathcal{I}'_{\mathcal{K},i+1}}$ and $d \in A_1^{\mathcal{I}'_{\mathcal{K},i+1}}$ by assumption.

  If $d \in \Delta^{\mathcal{I}^f_{\mathcal{K}}}$, then we must have $R \in \mathsf{N_{RR}}$, $(e, d) \in R^{\mathcal{I}'_{\mathcal{K},0}}$ and $(e, d) \in R^{\mathcal{I}^f_{\mathcal{K}}}$. By the assumption that $e \in B^{\mathcal{I}'_{\mathcal{K},i+1}} \setminus B^{\mathcal{I}'_{\mathcal{K},i}}$, we then get that $d \in A_1^{\mathcal{I}'_{\mathcal{K},i+1}} \setminus A_1^{\mathcal{I}'_{\mathcal{K},i}}$. From the previous case and the base cases for concept names $B \in \mathsf{N_{RC}}$, we know that there is a rigid concept D over $\mathcal{O}$ such that $d \in D^{\mathcal{I}^f_{\mathcal{K}}}$ and $\mathcal{O} \models D \sqsubseteq A_1$. But then $C := \exists R.D$ is as required since $e \in (\exists R.D)^{\mathcal{I}^f_{\mathcal{K}}}$ and $\mathcal{O} \models \exists R.D \sqsubseteq \exists R.A_1$.

  If $d = u'_{A_2} \in \Delta_{\mathsf{u}}^{\mathcal{I}'_{\mathcal{K}}}$, then Lemma 4.10 yields that $\mathcal{O} \models A_2 \sqsubseteq A_1$ and $u'_{A_2} \in A_1^{\mathcal{I}'_{\mathcal{K},0}}$. Since $e \in B^{\mathcal{I}'_{\mathcal{K},i+1}} \setminus B^{\mathcal{I}'_{\mathcal{K},i}}$, we know that there is a $B_1 \in \mathbb{B}(\mathcal{O})$ such that $e \in B_1^{\mathcal{I}'_{\mathcal{K},i}}$ and $\mathcal{O} \models B_1 \sqsubseteq \exists R.A_2$. By the induction hypothesis, there is a rigid concept over $\mathcal{O}$ which we choose to be C such that $e \in C^{\mathcal{I}^f_{\mathcal{K}}}$ and $\mathcal{O} \models C \sqsubseteq B_1$. Thus, we conclude that $\mathcal{O} \models C \sqsubseteq \exists R.A_1$, as required. □

4.4.2 Characterizing r-Satisfiability

In this section, we regard a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ as in Lemma 3.13, which consists of worlds $W_1, \dots, W_k$, and relates the LTL and the $\mathcal{EL}$ part of the satisfiability problem. We characterize the r-satisfiability of $\mathcal{W}$ as outlined in Section 3.3. Different from existing characterizations (see Section 3.2), we consider the exponentially many conjunctions from Definition 3.12 that are based on the worlds in $\mathcal{W}$ individually, so that we can apply the characterization in Algorithm 3.1: the idea is to specify a polynomial amount of additional data and conditions such that the former captures sufficient information about the rigid knowledge in the axioms induced by $W_1, \dots, W_k$ and is included into the conditions; the latter only regard the satisfiability of a single conjunction χ_i w.r.t. $\mathcal{O}$ for $i \in [1, k]$ at a time, can be tested using only polynomial space, and together characterize the satisfiability of the conjunctions in interpretations sharing a common domain (Function (F2)).

To synchronize the interpretations regarding the named individuals (Function (F1)), one part of the additional data are (negated) assertions of the relevant rigid names on all individual names occurring in Φ.

Definition 4.12 (ABox Type) An *ABox type* for Φ is a set

$$\begin{aligned}\mathcal{A}_{\mathsf{R}} \subseteq \{&A(a), \neg A(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\Phi),\ A \in \mathsf{N}_{\mathsf{RC}}(\mathcal{O})\} \cup \\ &\{R(a,b), \neg R(a,b) \mid a, b \in \mathsf{N}_{\mathsf{I}}(\Phi),\ R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O})\}\end{aligned}$$

with the property that $\alpha \in \mathcal{A}_{\mathsf{R}}$ iff $\neg\alpha \notin \mathcal{A}_{\mathsf{R}}$. ♢

Regarding Function (F2), we collect the (negated) axioms to be satisfied in ABoxes for all $i \in [1, k]$:

$$\mathcal{A}_{Q_i} := \{\varphi_j \mid p_j \in W_i\} \cup \{\neg\varphi_j \mid p_j \in \overline{W_i}\}.$$

The conditions to be satisfied for r-satisfiability can then be captured by the property of *r-completeness*.

Definition 4.13 (r-complete) A tuple $(\mathcal{A}_{\mathsf{R}}, \mathcal{A}^{\exists}_{\mathsf{R}})$ consisting of an ABox type $\mathcal{A}_{\mathsf{R}}$ for Φ and a set

$$\mathcal{A}^{\exists}_{\mathsf{R}} \subseteq \{\exists R.A(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\Phi),\ R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O}),\ A \in \mathsf{N}_{\mathsf{C}}(\mathcal{O}) \cup \{\top\}\}$$

is *r-complete* (w.r.t. $\mathcal{W}$ and Φ) if the following hold for all $i \in [1, k]$:

(C1) $\mathcal{K}^i_{\mathsf{R}} := \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}^{\exists}_{\mathsf{R}} \cup \mathcal{A}_{Q_i} \rangle$ is consistent.

(C2) For all $a \in \mathsf{N}_{\mathsf{I}}(\Phi)$, $R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O})$, and $A \in \mathsf{N}_{\mathsf{C}}(\mathcal{O}) \cup \{\top\}$, $a \in (\exists R.A)^{\mathcal{I}'_i}$ implies $\exists R.A(a) \in \mathcal{A}^{\exists}_{\mathsf{R}}$, where $\mathcal{I}'_i := \mathcal{I}'_{[\mathcal{K}^i_{\mathsf{R}}]^+}$, and $[\mathcal{K}^i_{\mathsf{R}}]^+$ is obtained from $\mathcal{K}^i_{\mathsf{R}}$ by dropping all negated assertions. ♢

The idea is that $\mathcal{A}_{\mathsf{R}}$ fixes the interpretation of the rigid names on the named individuals and that $\mathcal{A}^{\exists}_{\mathsf{R}}$ specifies what kind of successors need to be present at every time point, regarding all rigid roles. Recall that we originally consider the TKB to be empty when regarding satisfiability in $\mathcal{EL}$-LTL, and in the setting with global GCIs only modify this assumption w.r.t. the ontology. Since $\mathcal{W}$ is required to be non-empty, we thus can

disregard the empty ABox $\mathcal{A}_0$ from the TKB in our characterization by Fact 3.15. We specifically can show that the existence of an r-complete tuple w.r.t. $\mathcal{W}$ characterizes the r-satisfiability of this set.

Lemma 4.14 *$\mathcal{W}$ is r-satisfiable w.r.t. Φ iff there is an r-complete tuple w.r.t. $\mathcal{W}$ and Φ.*

The proof of this result is split over the subsequent sections.

If $\mathcal{W}$ is r-satisfiable, then there is an r-complete tuple w.r.t. $\mathcal{W}$ and Φ

Let $\mathcal{J}_1, \dots, \mathcal{J}_k$ be the interpretations that exist according to the r-satisfiability of $\mathcal{W}$. We construct ABoxes $\mathcal{A}_\mathsf{R}$ and $\mathcal{A}_\mathsf{R}^\exists$ as follows:

$$\begin{aligned}\mathcal{A}_\mathsf{R} := \{&A(a) \mid A \in \mathsf{N}_\mathsf{RC}(\mathcal{O}),\ a \in \mathsf{N}_\mathsf{I}(\Phi),\ \mathcal{J}_1 \models A(a)\} \cup \\ &\{\neg A(a) \mid A \in \mathsf{N}_\mathsf{RC}(\mathcal{O}),\ a \in \mathsf{N}_\mathsf{I}(\Phi),\ \mathcal{J}_1 \not\models A(a)\} \cup \\ &\{R(a,b) \mid R \in \mathsf{N}_\mathsf{RR}(\mathcal{O}),\ a,b \in \mathsf{N}_\mathsf{I}(\Phi),\ \mathcal{J}_1 \models R(a,b)\} \cup \\ &\{\neg R(a,b) \mid R \in \mathsf{N}_\mathsf{RR}(\mathcal{O}),\ a,b \in \mathsf{N}_\mathsf{I}(\Phi),\ \mathcal{J}_1 \not\models R(a,b)\},\end{aligned}$$

$$\begin{aligned}\mathcal{A}_\mathsf{R}^\exists := \{&\exists R.A(a) \mid R \in \mathsf{N}_\mathsf{RR}(\mathcal{O}),\ A \in \mathsf{N}_\mathsf{C}(\mathcal{O}) \cup \{\top\},\ a \in \mathsf{N}_\mathsf{I}(\Phi), \\ &C \text{ a rigid concept over } \mathcal{O},\ a \in C^{\mathcal{J}_1},\ \mathcal{O} \models C \sqsubseteq \exists R.A\}.\end{aligned}$$

In what follows, we focus on Definition 4.13 and show that the tuple $(\mathcal{A}_\mathsf{R}, \mathcal{A}_\mathsf{R}^\exists)$ is r-complete. The set $\mathcal{A}_\mathsf{R}$ is obviously an ABox type and, since the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$ agree on the interpretation of all rigid names by assumption, each of them is a model of $\mathcal{A}_\mathsf{R}$. Furthermore, these interpretations satisfy $\mathcal{O}$ and hence also $\mathcal{A}_\mathsf{R}^\exists$. This implies that, for all $i \in [1,k]$, $\mathcal{K}_\mathsf{R}^i = \langle \mathcal{O}, \mathcal{A}_\mathsf{R} \cup \mathcal{A}_\mathsf{R}^\exists \cup \mathcal{A}_{Q_i} \rangle$ is consistent.

Regarding the other condition of r-completeness, we assume that $a \in (\exists R.A)^{\mathcal{I}_i'}$ holds for some $a \in \mathsf{N}_\mathsf{I}(\Phi)$, $R \in \mathsf{N}_\mathsf{RR}(\mathcal{O})$, $A \in \mathsf{N}_\mathsf{C}(\mathcal{O}) \cup \{\top\}$, and $i \in [1,k]$, and show that we then have $\exists R.A \in \mathcal{A}_\mathsf{R}^\exists$. By Lemma 4.11, there is a rigid concept C over $\mathcal{O}$ such that $a \in C^{\mathcal{I}_i}$ and $\mathcal{O} \models C \sqsubseteq \exists R.A$, where $\mathcal{I}_i := \mathcal{I}_{[\mathcal{K}_\mathsf{R}^i]^+}$ is the finite canonical interpretation for $[\mathcal{K}_\mathsf{R}^i]^+$. From Lemma 2.17, we then get that $a \in C^{\mathcal{J}_i}$, and thus $a \in C^{\mathcal{J}_1}$ since $\mathcal{J}_i$ and $\mathcal{J}_1$ agree on the interpretation of the rigid names.

It remains to prove the other direction of Lemma 4.14.

If there is an r-complete tuple w.r.t. $\mathcal{W}$ and Φ, then $\mathcal{W}$ is r-satisfiable w.r.t. Φ

We focus on an r-complete tuple $(\mathcal{A}_\mathsf{R}, \mathcal{A}_\mathsf{R}^\exists)$ and in the following first provide auxiliary definitions, then define interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$, and lastly show that these interpretations satisfy the requirements stated in Definition 3.12. The idea of our construction of the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$ is roughly to integrate the finite canonical interpretations of the KBs in Condition (C1) of Definition 4.13. This can be done by using the corresponding rigid canonical interpretations to include consequences of rigid information from the KBs, which are invariant to time. For all $i \in [1,k]$, consider the following definitions.

- Let $\mathcal{I}_i := \mathcal{I}_{[\mathcal{K}_\mathsf{R}^i]^+}$ be the finite canonical interpretation of the KB $[\mathcal{K}_\mathsf{R}^i]^+$ obtained from $\mathcal{K}_\mathsf{R}^i$ by removing all negated assertions from $\mathcal{A}_\mathsf{R}$ and $\mathcal{A}_{Q_i}$; and let $\mathcal{I}_i' := \mathcal{I}'_{[\mathcal{K}_\mathsf{R}^i]^+}$ be the corresponding rigid canonical interpretation.

We show that $\mathcal{I}_i$ is a model of $\mathcal{K}^i_{\mathsf{R}}$, though. Since $\mathcal{I}_i$ satisfies $[\mathcal{K}^i_{\mathsf{R}}]^+$ by Lemma 2.12, we can focus on the negated assertions in $\mathcal{A}_{\mathsf{R}}$ and $\mathcal{A}_{Q_i}$. Given that $\mathcal{K}^i_{\mathsf{R}}$ is consistent by Condition (C1), we know that $[\mathcal{K}^i_{\mathsf{R}}]^+ \not\models \alpha$ holds for all negated assertions $\neg\alpha \in \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}^{\exists}_{\mathsf{R}}$. By Definition 2.10, it then follows that $\mathcal{I}_i \models \neg A(a)$, which implies the following.

Fact 4.15 *$\mathcal{I}_i$ is a model of $\mathcal{K}^i_{\mathsf{R}}$.*

- We write $\Delta_{\mathsf{u}}^{\mathcal{I}_i}$ to distinguish the unnamed domain elements unique to the canonical interpretation $\mathcal{I}_i$ and u^i_A instead of u_A for the elements in this set.

  Thus, the domain of $\mathcal{I}_i$ is composed of the pairwise disjoint sets $\mathsf{N}_{\mathsf{I}}(\Phi)$ and $\Delta_{\mathsf{u}}^{\mathcal{I}_i}$.

  Fact 4.16 *The set $\mathsf{N}_{\mathsf{I}}(\Phi)$ and all sets $\Delta_{\mathsf{u}}^{\mathcal{I}_i}$ with $i \in [1,k]$ are pairwise disjoint.*

We now construct the interpretations $\mathcal{J}_1, \ldots, \mathcal{J}_k$ as required to show the r-satisfiability of $\mathcal{W}$ by joining the domains of the interpretations $\mathcal{I}_i$ and ensuring that they interpret all rigid names in the same way. The common domain Δ is specified as follows:

$$\Delta := \mathsf{N}_{\mathsf{I}}(\Phi) \cup \bigcup_{i=1}^{k} \Delta_{\mathsf{u}}^{\mathcal{I}_i}.$$

$\mathcal{J}_i$ is defined below, for all $i \in [1,k]$:

- For all $a \in \mathsf{N}_{\mathsf{I}}(\Phi)$: $a^{\mathcal{J}_i} := a$.
- For all rigid concept names A: $A^{\mathcal{J}_i} := \bigcup_{j=1}^{k} A^{\mathcal{I}_j}$.
- For all flexible concept names A: $A^{\mathcal{J}_i} := A^{\mathcal{I}_i} \cup \bigcup_{j=1}^{k} \left(\Delta_{\mathsf{u}}^{\mathcal{I}_j} \cap A^{\mathcal{I}'_j}\right)$.
- For all rigid role names R:

$$R^{\mathcal{J}_i} := \bigcup_{j=1}^{k} R^{\mathcal{I}_j} \cup \bigcup_{j=1}^{k} \bigcup_{\ell=1}^{k} \{(e, u^{\ell}_A) \mid\ e \in \Delta_{\mathsf{u}}^{\mathcal{I}_j} \cap (\exists R.A)^{\mathcal{I}'_j}\}.$$

- For all flexible role names R:

$$R^{\mathcal{J}_i} := R^{\mathcal{I}_i} \cup \bigcup_{j=1}^{k} \{(e, u^i_A) \mid e \in \Delta_{\mathsf{u}}^{\mathcal{I}_j} \cap (\exists R.A)^{\mathcal{I}'_j}\}.$$

We thus have constructed interpretations $\mathcal{J}_1, \ldots, \mathcal{J}_k$ that share one domain, respect rigid names, and satisfy the UNA for all relevant individual names. It remains to show that each $\mathcal{J}_i$ is a model of $\mathcal{O}$ and χ_i. To this end, we first characterize it in terms of the canonical interpretations it is based upon.

Lemma 4.17 *For all $i, j \in [1,k]$ and concepts $B \in \mathbb{S}(\mathcal{O})$, the following hold:*

a) *For all $a \in \mathsf{N}_{\mathsf{I}}(\Phi)$, we have $a \in B^{\mathcal{J}_i}$ iff $a \in B^{\mathcal{I}_i}$.*

b) *For all* $e \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$*, we have* $e \in B^{\mathcal{J}_i}$ *iff*

- $i = j$ *and* $e \in B^{\mathcal{I}_i}$*, or*
- $e \in B^{\mathcal{I}'_j}$.

Proof. Regarding b), observe first that $e \in B^{\mathcal{I}'_i}$ implies $e \in B^{\mathcal{I}_i}$ by Lemma 4.11 and the fact that $\mathcal{I}_i$ is a model of $\mathcal{O}$. This means that, if $i = j$, then the two items are equivalent to the first item. On the other hand, if $i \neq j$, then only the second item in b) has to be considered. We prove a) and b) simultaneously by induction on the structure of B and start with the base cases. The claims obviously hold for $B = \top$ and, for a flexible concept name B, they follow directly from the definition of $\mathcal{J}_i$ and Fact 4.16.

Let $B \in \mathsf{N_{RC}}(\mathcal{O})$. For a), we have $a \in B^{\mathcal{J}_i}$ iff there is some $j \in [1, k]$ such that $a \in B^{\mathcal{I}_j}$ by the definition of $\mathcal{J}_i$. Since both $\mathcal{I}_i$ and $\mathcal{I}_j$ are models of the ABox type $\mathcal{A}_{\mathsf{R}}$, this is equivalent to $a \in B^{\mathcal{I}_i}$.

For b) and $i = j$, Fact 4.16 and the definition of $\mathcal{J}_i$ yield the claim. For $i \neq j$, we additionally observe that $B^{\mathcal{I}'_j} \cap \Delta^{\mathcal{I}_j} = B^{\mathcal{I}_j}$ holds by Definition 4.9.

Regarding the induction steps, we skip the case for $B = A_1 \sqcap A_2$ since it can be easily treated based on the semantics and assume $B = \exists R.A$.

($\Leftarrow$) For both a) and b), this direction is a direct consequence of the observations that $R^{\mathcal{I}_i} \subseteq R^{\mathcal{J}_i}$ and $A^{\mathcal{I}_i} \subseteq A^{\mathcal{J}_i}$; regarding the last item, we additionally mention that $e \in (\exists R.A)^{\mathcal{I}'_j}$ implies $(e, u_A^i) \in R^{\mathcal{J}_i}$ and $u_A^i \in A^{\mathcal{J}_i}$ by the definition of $\mathcal{J}_i$ and, w.r.t. the latter, Definitions 2.10 and 4.9.

($\Rightarrow$) We begin with the proof of a). If R is rigid, then the definition of $\mathcal{J}_i$ implies that there is an element $e \in \Delta^{\mathcal{I}_j}$, $j \in [1, k]$, such that $(a, e) \in R^{\mathcal{I}_j}$ and $e \in A^{\mathcal{J}_i}$.

- If $e \in \mathsf{N_I}(\Phi)$, then we have $R(a, e) \in \mathcal{A}_j \cup \mathcal{A}_{\mathsf{R}}$ by Definition 2.10. Since $\mathcal{K}_{\mathsf{R}}^j$ is consistent and $\mathcal{A}_{\mathsf{R}}$ is an ABox type, this yields that $R(a, e) \in \mathcal{A}_{\mathsf{R}}$ and leads to $(a, e) \in R^{\mathcal{I}_i}$ by Lemma 2.12. $a \in (\exists R.A)^{\mathcal{I}_i}$ is obtained by the induction hypothesis w.r.t. e.

- If $i = j$ and $e \in \Delta_{\mathsf{u}}^{\mathcal{I}_i}$, then the induction hypothesis yields $e \in A^{\mathcal{I}_i}$, which leads to $a \in (\exists R.A)^{\mathcal{I}_i}$.

- It remains to consider the case that $i \neq j$ and $e \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$. Then, we have $(a, e) \in R^{\mathcal{I}'_j}$ by Definition 4.9. From the induction hypothesis, we obtain $e \in A^{\mathcal{I}'_j}$, and hence $a \in (\exists R.A)^{\mathcal{I}'_j}$. Thus, we have $\exists R.A(a) \in \mathcal{A}_{\mathsf{R}}^{\exists}$, and hence $a \in (\exists R.A)^{\mathcal{I}_i}$ since $\mathcal{I}_i$ is a model of $\mathcal{A}_{\mathsf{R}}^{\exists}$.

If R is flexible, then there is an element $e \in \Delta^{\mathcal{I}_i}$ such that $(a, e) \in R^{\mathcal{I}_i}$ and $e \in A^{\mathcal{J}_i}$ by the definition of $\mathcal{J}_i$ and Fact 4.16. By applying the induction hypothesis to e (and $i = j$), we obtain that $e \in A^{\mathcal{I}_i}$ and thus $a \in (\exists R.A)^{\mathcal{I}_i}$.

For b), we begin with the case that R is rigid. By the definition of $\mathcal{J}_i$, either (i) there is an element d such that $(e, d) \in R^{\mathcal{I}_j}$ and $d \in A^{\mathcal{J}_i} \cap \Delta_{\mathsf{u}}^{\mathcal{I}_j}$ (see Definition 2.10 and Fact 4.16), or (ii) $e \in (\exists R.A)^{\mathcal{I}'_j}$, $(e, u_A^\ell) \in R^{\mathcal{J}_i}$, and $u_A^\ell \in A^{\mathcal{J}_i}$ for some $\ell \in [1, k]$ (again by Fact 4.16). By the initial observation, we thus only have to consider (i). By the induction hypothesis, we have either (i') $i = j$ and $d \in A^{\mathcal{I}_i}$ or (ii') $d \in A^{\mathcal{I}'_j}$. In the first case, we can infer that $e \in (\exists R.A)^{\mathcal{I}_i}$ and, in the second case, we have $(e, d) \in R^{\mathcal{I}'_j}$ since R is rigid, and thus $e \in (\exists R.A)^{\mathcal{I}'_j}$.

If R is flexible, then either (i) there is an element d such that $(e,d) \in R^{\mathcal{I}_i}$ and $d \in A^{\mathcal{J}_i} \cap \Delta_{\mathsf{u}}^{\mathcal{I}_i}$ (see Definition 2.10 and Fact 4.16), or (ii) $e \in (\exists R.A)^{\mathcal{I}'_j}$, $(e, u_A^i) \in R^{\mathcal{J}_i}$, and $u_A^i \in A^{\mathcal{J}_i}$. Again, Case (ii) is trivial. In Case (i), we have $i = j$, and thus by the induction hypothesis get that $d \in A^{\mathcal{I}_i}$. Thus, we get $e \in (\exists R.A)^{\mathcal{I}_i}$. □

Based on this lemma, we show that the interpretations $\mathcal{J}_i$ are in fact as intended.

Lemma 4.18 *For all $i \in [1,k]$, $\mathcal{J}_i$ is a model $\mathcal{O}$.*

Proof. We consider a GCI $B \sqsubseteq C \in \mathcal{O}$ and arbitrary interpretation $\mathcal{J}_i$ with $i \in [1,k]$. Let first $e \in \Delta^{\mathcal{I}_i} \cap B^{\mathcal{J}_i}$. By Lemma 4.17, we have $e \in B^{\mathcal{I}_i}$, and hence get $e \in C^{\mathcal{I}_i}$ since $\mathcal{I}_i$ satisfies $\mathcal{O}$. By again applying Lemma 4.17, we obtain $e \in C^{\mathcal{J}_i}$.

It remains to consider $e \in \Delta_{\mathsf{u}}^{\mathcal{I}_j} \cap B^{\mathcal{J}_i}$ for $i \neq j$. By Lemma 4.17, we get $e \in B^{\mathcal{I}'_j}$. Next, we show that $e \in C^{\mathcal{I}'_j}$ then holds, which leads to $e \in C^{\mathcal{J}_i}$, again by Lemma 4.17. Since $e \in B^{\mathcal{I}'_j}$, there is an $\ell \geq 0$ such that $e \in B^{\mathcal{I}'_{j,\ell}}$, where $\mathcal{I}'_{j,\ell} := \mathcal{I}'_{[\mathcal{K}_{\mathsf{R}}^j]^+,\ell}$ is as in Definition 4.9.

- Let $C = \top$. In this case, $e \in C^{\mathcal{I}'_j}$ holds trivially by the semantics.
- Let $C \in \mathsf{N_C} \setminus \mathsf{N_{RC}}$. $e \in B^{\mathcal{I}'_{j,\ell}}$ and the GCI yield $e \in C^{\mathcal{I}'_{j,\ell+1}}$ and $C^{\mathcal{I}'_{j,\ell+1}} \subseteq C^{\mathcal{I}'_j}$ by the definitions of $\mathcal{I}'_{j,\ell+1}$ and $\mathcal{I}'_j$.
- $C \in \mathsf{N_{RC}}$. $e \in B^{\mathcal{I}'_j}$ implies $e \in B^{\mathcal{I}_j}$ by Lemma 4.11 and the fact that $\mathcal{I}_j$ is a model of $\mathcal{O}$. We also have $B^{\mathcal{I}_j} \subseteq C^{\mathcal{I}_j}$ by Lemma 2.12, $C^{\mathcal{I}_j} \subseteq C^{\mathcal{I}'_{j,0}}$ by the kind of e and the definition of $\mathcal{I}'_{j,0}$, and $C^{\mathcal{I}'_{j,0}} \subseteq C^{\mathcal{I}'_j}$ by Definition 4.9. We thus have $e \in C^{\mathcal{I}'_j}$.
- Let $C = \exists R.A$. $e \in B^{\mathcal{I}'_{j,\ell}}$ and the GCI yield $(e, u'_A) \in R^{\mathcal{I}'_{j,\ell+1}}$, which is a subset of $R^{\mathcal{I}'_j}$ by the definition of $\mathcal{I}'_{j,\ell+1}$. Since we also have $u'_A \in A^{\mathcal{I}'_{j,0}}$, which is a subset of $A^{\mathcal{I}'_j}$, we obtain $e \in (\exists R.A)^{\mathcal{I}'_j}$. □

It remains to show that all literals in χ_i are satisfied by $\mathcal{J}_i$, which are the (negated) assertions $\mathcal{A}_{Q_i}$.

Lemma 4.19 *For all $i \in [1,k]$, $\mathcal{J}_i$ is a model of χ_i.*

Proof. For the positive assertions in $\mathcal{A}_{Q_i}$, we know that $\mathcal{I}_i$ satisfies $\mathcal{A}_{Q_i}$, by Definition 4.13 and Lemma 2.12. Since we have $A^{\mathcal{I}_i} \subseteq A^{\mathcal{J}_i}$ and $R^{\mathcal{I}_i} \subseteq R^{\mathcal{J}_i}$ for all $A \in \mathsf{N_C}(\mathcal{O})$ and $R \in \mathsf{N_R}(\mathcal{O})$, these assertions from $\mathcal{A}_{Q_i}$ are also satisfied in $\mathcal{J}_i$.

We regard a negated concept assertion $\neg A(a) \in \mathcal{A}_{Q_i}$. Since $\mathcal{I}_i \models \mathcal{A}_{Q_i}$, we have $a \notin A^{\mathcal{I}_i}$, and thus $a \notin A^{\mathcal{J}_i}$ follows from Lemma 4.17.

For a negated assertion $\neg R(a,b) \in \mathcal{A}_{Q_i}$ with $R \in \mathsf{N_R} \setminus \mathsf{N_{RR}}$, we similarly get $(a,b) \notin R^{\mathcal{I}_i}$, and thus $(a,b) \notin R^{\mathcal{J}_i}$ follows from the definition of $\mathcal{J}_i$.

Regarding a negated assertion $\neg R(a,b) \in \mathcal{A}_{Q_i}$ with $R \in \mathsf{N_{RR}}$, we get $\neg R(a,b) \in \mathcal{A}_{\mathsf{R}}$ from the consistency of $\mathcal{K}_{\mathsf{R}}^i$ and the fact that $\mathcal{A}_{\mathsf{R}}$ is an ABox type. Thus, Fact 4.15 implies $(a,b) \notin R^{\mathcal{I}_j}$ for all $j \in [1,k]$, and we get $(a,b) \notin R^{\mathcal{J}_i}$, again by the definition of $\mathcal{J}_i$. □

This concludes the proof of Lemma 4.14.

4.4.3 Containment

We finally show that the satisfiability problem in $\mathcal{EL}$-LTL with global GCIs is in PSPACE, even if rigid symbols are considered. The characterization of r-satisfiability from the previous section allows us to integrate corresponding tests with the t-satisfiability testing as described in Algorithm 3.1. Recall that the latter is based upon the polynomial-space algorithm originally proposed for deciding satisfiability in LTL (see Algorithm 2.1). In what follows, we specify our approach and show that the (nondeterministic) r-satisfiability tests can be done by using only polynomial space, which yields that it is as required.

Definition 4.20 Given a formula $\bigwedge \Phi_{\mathcal{O}} \wedge \Phi$ in $\mathcal{EL}$-LTL with global GCIs, satisfiability can be decided by running Algorithm 3.1 with input Φ and $\langle \mathcal{O}, \emptyset \rangle$:

- `GUESSDATA`: It guesses and returns a tuple $(\mathcal{A}_{\mathsf{R}}, \mathcal{A}_{\mathsf{R}}^{\exists})$ consisting of an ABox type $\mathcal{A}_{\mathsf{R}}$ for $\bigwedge \Phi_{\mathcal{O}} \wedge \Phi$ and a set

$$\mathcal{A}_{\mathsf{R}}^{\exists} \subseteq \{\exists R.A(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\Phi),\ R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O}),\ A \in \mathsf{N}_{\mathsf{C}}(\mathcal{O}) \cup \{\top\}\}.$$

- `TESTRSAT`: Given $\Phi, \mathcal{O}, \mathcal{A}_i, d$ (i.e., the tuple guessed), i, s, p, and W, it defines

$$\mathcal{A}_{Q_i} := \{\varphi_j \mid p_j \in W\} \cup \{\neg\varphi_j \mid p_j \in \overline{W}\}$$

  and returns *true* iff the two conditions in Definition 4.13 are satisfied. ♢

Observe that, since we have $n = 0$, our algorithm adapts Algorithm 2.1 only in the guessing and r-satisfiability testing.

Lemma 4.21 *The nondeterministic algorithm described in Definition 4.20 decides satisfiability in $\mathcal{EL}$-LTL with global GCIs by using only polynomial space (in the size of the given formula).*

Proof. Let the set $\mathcal{W}$ be defined as union of all sets W encountered while running the algorithm. Then, the correctness follows from Lemma 3.13; Fact 3.15; the correctness of the LTL algorithm (see Lemma 2.21) regarding the satisfiability of Φ^{pa}; the observation that we only extend the latter by the guessing, which does not influence correctness, and the r-satisfiability testing; and Lemma 4.14 together with the fact that our adaptation leads to a negative result iff one of the conditions of r-completeness is not satisfied.

The space complexity is a consequence of Lemma 2.21 about Algorithm 2.1, the fact that we only extend the latter by the guessing and the r-satisfiability parts, and the following observations about them.

- The nondeterministic guessing of the polynomially large sets $\mathcal{A}_{\mathsf{R}}$ and $\mathcal{A}_{\mathsf{R}}^{\exists}$ can be done using only polynomial space.
- The KB considered in the conditions checked can be seen as a conjunction of $\mathcal{EL}$ literals (of size polynomial in that of Φ and $\mathcal{O}$). Thus, its consistency can be decided in deterministic polynomial time by Lemma 4.1.
- The size of the rigid canonical interpretation considered in Condition (C2) is polynomial in the size of $\mathcal{O}$ (see Definition 4.9). Regarding one such interpretation, that condition can hence be checked in polynomial time. □

				Global GCIs		
$\mathcal{DL}$	(i)	(ii)	(iii)	(i)	(ii)	(iii)
TDL-Lite$^{\bigcirc_F}_{krom}$ [a]	PSPACE	PSPACE	PSPACE	PSPACE	PSPACE	PSPACE
TDL-Lite$^{\bigcirc_F}_{horn}$ [a]	EXPSPACE	EXPSPACE	EXPSPACE	EXPSPACE	EXPSPACE	EXPSPACE
TDL-Lite$_{bool}$ [a]	EXPSPACE	EXPSPACE	EXPSPACE	EXPSPACE	EXPSPACE	EXPSPACE
$\mathcal{EL}$	PSPACE	NEXPTIME	NEXPTIME	PSPACE	PSPACE	PSPACE
	$\geq$ [SC85], $\leq$ Co. 4.5	$\geq$ Th. 4.8	$\leq$ Co. 4.6			$\leq$ Co. 4.22
DL-Lite$^{[\,|\mathcal{H}]}_{horn}$	PSPACE	NEXPTIME	NEXPTIME	PSPACE	PSPACE	PSPACE
	$\geq$ [SC85], $\leq$ Co. 4.5	$\geq$ Th. 4.8	$\leq$ Co. 4.6			$\leq$ Co. 6.19
DL-Lite$^{[\,|\mathcal{H}]}_{bool}$	PSPACE	NEXPTIME	NEXPTIME	PSPACE	$\leq$NEXPTIME	$\leq$NEXPTIME
	$\geq$ [SC85], $\leq$ Co. 4.5	$\geq$ Th. 4.8	$\leq$ Co. 4.6			
$\mathcal{ALC}$ [b]	EXPTIME	NEXPTIME	2-EXPTIME	EXPTIME	EXPTIME	2-EXPTIME
$\mathcal{SHOQ}$ [c b]	EXPTIME	NEXPTIME	2-EXPTIME	EXPTIME	$\leq$NEXPTIME	2-EXPTIME

Figure 4.1: The complexity of satisfiability in $\mathcal{DL}$-LTL for different DLs $\mathcal{DL}$ considering (i) no rigid symbols, (ii) rigid concept names, and (iii) rigid role names.[d] Our results are highlighted. All complexities except those marked with $\leq$ are tight; $\geq$ hardness, $\leq$ containment.

[a] [Art+07]
[b] [BGL12]
[c] [Lip14]
[d] The temporal DLs from the related works are described in Section 3.4.

The complexity of satisfiability in $\mathcal{EL}$-LTL with global GCIs and in the presence of rigid symbols is then obtained by combining Lemma 4.21 and the well-known result of Savitch [Sav70, Thm. 1].

Corollary 4.22 *Satisfiability in $\mathcal{EL}$-LTL with global GCIs is in* PSPACE*, even in case that* $\mathsf{N_{RR}} \neq \emptyset$.

4.5 Summary

In this chapter, we have investigated the complexity of satisfiability in several temporal lightweight DLs allowing for temporalizing DL axioms. We specifically covered $\mathcal{EL}$-LTL as well as all logics between *DL-Lite*$_{horn}$-LTL and *DL-Lite*$^{\mathcal{H}}_{bool}$-LTL, and different settings regarding the rigid symbols allowed. Although results for *TDL-Lite*$_{bool}$-LTL [Art+07] and $\mathcal{ALC}$-LTL [BGL12] have been obtained before, even decidability was still open for the logics including role hierarchies. We have shown decidability, PSPACE containment for the case without rigid symbols, and that the interaction of the LTL and DL features strongly increases with rigid symbols—although we considered very small DLs.

Figure 4.1 shows the results in comparison to formalisms that similarly allow for temporalizing DL axioms using LTL. Compared to *TDL-Lite*$^{\bigcirc_F}_{krom}$-LTL—which subsumes *DL-Lite*$_{krom}$-LTL—, there is a considerable increase in complexity if rigid symbols are considered, already w.r.t. concept names. Note that we also have this NEXPTIME-

completeness for $\mathcal{SHOQ}$-LTL, which offers much more expressivity, if inverse roles are disregarded. On the other hand, the restriction to global GCIs and basic concept assertions has turned out as surprisingly beneficial for $\mathcal{EL}$-LTL, given the corresponding results for $\mathcal{ALC}$-LTL. The 2-ExpTime complexity regarding $\mathcal{SHOQ}$-LTL with global GCIs follows from the case without restrictions and the result for $\mathcal{ALC}$-LTL with global GCIs. Note that the results for global GCIs regarding the Horn fragment of *DL-Lite* follow from those for TCQs, presented in Chapter 6; that is, $\mathcal{DL}$-LTL formulas with global GCIs can be seen as TCQs if the GCIs are taken as the global ontology. For the logics in the Bool fragment of *DL-Lite*, we have not shown tight complexity bounds. We do not estimate them to be NExpTime-complete since we need more expressivity in all the NExpTime-hardness proofs in this work.

Furthermore, it should be noted that $\mathcal{EL}$-LTL represents a temporal version of one of the DLs that are most important in practice and for which there are many ontologies around, which basically are global GCIs. Our results show, what was unknown given only the results for *DL-Lite*-LTL or $\mathcal{ALC}$-LTL: formulas combining $\mathcal{EL}$ basic concept and role assertions via LTL operators can be considered w.r.t. $\mathcal{EL}$ ontologies "for free"—compared to formulas in LTL. This proposes two alternatives for practically applying temporalized DL axioms: to disallow rigid symbols or to consider the CIs as being global.

5 Temporal Query Entailment in $\mathcal{EL}$

In this chapter, we regard a Boolean TCQ Φ and a TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ in $\mathcal{EL}$ and investigate the combined and data complexity of the TCQ entailment problem. We show that, leaving rigid symbols aside, neither combined nor data complexity increase compared to the respective baselines, the combined complexity of satisfiability in LTL and the data complexity of CQ answering in $\mathcal{EL}$. The former even holds if rigid concept names are considered. Regarding data complexity, tractability is however lost if rigid names are taken into account. Moreover, the co-NExpTime combined complexity we show for the case with rigid roles does not seem to be attractive for most applications of $\mathcal{EL}$.

The considerable influence of rigid symbols is due to the main expressive feature of $\mathcal{EL}$, qualified existential restriction (in combination with conjunction)—this gets evident when we consider $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$ in Chapter 6, which only allows existential restrictions to be unqualified. For example, $\mathcal{EL}$ ontologies may contain GCIs as the following, stating that every element which has a system-critical part that is defect has a critical defect:

$$\exists \mathtt{HasPart}.\mathtt{Defect} \sqcap \mathtt{SystemCritical} \sqsubseteq \mathtt{CriticalDefect}.$$

Hence, if the role `HasPart` is rigid and used to connect components, then the states of the latter, at any time point i, depend not only on the states of all their subcomponents at i but at arbitrary points in time. Observe also that (T)CQs as the following similarly allow to query for such knowledge, but the answers then cannot be taken into account during inferencing:

$$\exists y.\mathtt{HasPart}(x, y) \land \mathtt{Defect}(y) \land \mathtt{SystemCritical}(y).$$

TCQ satisfiability—and hence entailment—can be decided by solving two satisfiability problems, according to Lemma 3.13: one in LTL and one in DL; the latter is captured by the notion of r-satisfiability. Our results are based on this approach. The set $\mathcal{W}$ of LTL worlds which is to be found, connects the two problems, and plays an important role in both is however of exponential size. In Section 5.1, we therefore first propose a new characterization of r-satisfiability in the presence of rigid concept names that is tailored to TCQs and $\mathcal{EL}$ and, different from existing characterizations (see Section 3.2), is not inherently of exponential complexity. We then apply this approach in the subsequent Section 5.2 on combined complexity for obtaining the PSpace result. Data complexity is investigated in Section 5.3.

Throughout the chapter, we use the notation of Section 3.2. For simplicity, we further assume $\mathcal{K}$ to be of the form $\mathcal{K} = \langle \mathcal{O}, \emptyset \rangle$ where we target combined complexity (i.e., in Sections 5.1 and 5.2), which can be done according to Lemma 3.7. Observe that entailment is then decided by focusing on time point zero. With this assumption in place, we however have to drop the assumption that all individual names contained in Φ

also occur in the ABoxes; in fact, Φ is then the only place where individual names may occur.

We close the introduction with an auxiliary result on the satisfiability of conjunctions of CQ literals. The latter represents an integral problem in the existing characterizations of r-satisfiability, which we apply for containment results.

Satisfiability of Conjunctions of CQ literals

We show that, for a Boolean TCQ that is a conjunction of CQ literals, the satisfiability w.r.t. a classical KB can be decided in P w.r.t. combined complexity. The proof is by reduction to UCQ non-entailment through an instantiation of the positive literals, similar to the proof of the corresponding Theorem 4.1 in [BBL15b]—that theorem considers the DL $\mathcal{SHQ}$ and targets another complexity class—, and a careful analysis of a procedure to solve that problem in NP, which has been proposed in [Ros07].

Lemma 5.1 *For a knowledge base $\mathcal{K}$ and a Boolean conjunction Ψ of CQ literals, the decision if Ψ has a model w.r.t. $\mathcal{K}$ can be reduced to several deterministic polynomial time tests w.r.t. combined complexity, the number of which is polynomial in the number of conjuncts of Ψ and exponential in the maximal number of terms occurring in a negative CQ literal in Ψ.*

Proof. Let $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$. As it is done in the proof of the corresponding Theorem 4.1 in [BBL15b], we first reduce the problem of deciding whether Ψ has a model w.r.t. $\mathcal{K}$ to a UCQ non-entailment problem. Let

$$\Psi = \varphi_1 \wedge \cdots \wedge \varphi_\ell \wedge \neg\psi_1 \wedge \ldots \wedge \neg\psi_m$$

where $\varphi_1, \ldots, \varphi_\ell, \psi_1, \ldots, \psi_m$ are Boolean CQs. Then, the positive CQ literals $\varphi_1, \ldots, \varphi_\ell$ are instantiated by omitting the existential quantifiers and replacing the variables by fresh individual names. The resulting set $\mathcal{A}'$ of assertions is then regarded as an additional ABox restricting possible models of $\mathcal{K}$.

It is easy to see that Ψ is satisfiable w.r.t. $\mathcal{K}$ iff there is an interpretation $\mathcal{I}'$ such that $\mathcal{I}' \models \langle \mathcal{O}, \mathcal{A} \cup \mathcal{A}' \rangle$ and $\mathcal{I}' \models \neg\psi_1 \wedge \ldots \wedge \neg\psi_m$. This is the complement of the entailment problem $\langle \mathcal{O}, \mathcal{A} \cup \mathcal{A}' \rangle \models \psi_1 \vee \ldots \vee \psi_m$.

In [Ros07, Thm. 2], it is shown that the latter problem is NP-complete w.r.t. combined complexity, which seems to contradict our above claim. The proof is based on an algorithm (`computeQueryEntailment`) that decides UCQ entailment. In particular, it is remarked in [Ros07] that the nondeterminism is caused only by the first step of the algorithm, while all other steps run in deterministic polynomial time in their input. This first step (sub-procedure `unify`) nondeterministically chooses one CQ ψ_i with $i \in [1, m]$ and one substitution to unify some of the terms in ψ_i. But this means that we can instead consider all (exponentially many) possible unifiers, for each ψ_i with $i \in [1, m]$, and execute the remaining deterministic steps of the algorithm `computeQueryEntailment`, for each of them, in polynomial time. In analogy to `computeQueryEntailment`, the entailment holds iff one of these runs succeeds.

Consequently, also the complement problem, satisfiability of Boolean UCQs, can be decided deterministically by applying exponentially many (in the size of the largest negated conjunct in Ψ) polynomial time tests. □

5.1 Characterizing r-Satisfiablility Without Rigid Roles

In this section, we regard a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ that consists of worlds $W_1, \dots, W_k$ and connects the LTL part of the TCQ satisfiability problem, looking for a model of the LTL formula Φ^{pa}, to the $\mathcal{EL}$ part, according to Lemma 3.13. The $\mathcal{EL}$ part is captured by the notion of r-satisfiability of $\mathcal{W}$, for which we in the following propose a new characterization. Our approach is outlined in Section 3.3. The idea is to specify a polynomial amount of additional data that allows us to split the tests proposed in Definition 3.12 into separate and independent consistency tests that focus on single elements of $\mathcal{W}$, may take this data into account, and require only polynomial space. We thus proceed similar as for $\mathcal{EL}$-LTL with global GCIs (see Section 4.4.2); but the conditions we develop for capturing the satisfiability of the conjunctions of CQ literals in the presence of rigid knowledge present a major difference.

The focus is on the set $\mathcal{W}$ fixed above; we can disregard the mapping ι and the last condition in Definition 3.12 by Fact 3.15. Note that, for now, we assume the entire set $\mathcal{W}$ to be given. We however never consider it as a whole, but regard its elements $W_1, \dots, W_k$ independently of each other. In what follows, we first describe the form of the additional data and then present the conditions to be tested.

Recall that the goal of including the additional data is to simulate the effects of the shared domain, Functions (F1) and (F2). To synchronize the interpretations regarding the named individuals (Function (F1)), the additional data contains an ABox type similar to the one for $\mathcal{EL}$-LTL with global GCIs (see Definition 4.12). But we disregard rigid roles here.

Definition 5.2 (ABox Type) An *ABox type* for $\mathcal{O}$ is a set

$$\mathcal{A}_{\mathsf{R}} \subseteq \{A(a), \neg A(a) \mid a \in \mathsf{N_I}(\Phi),\ A \in \mathsf{N_{RC}}(\mathcal{O})\}$$

with the property that $A(a) \in \mathcal{A}_{\mathsf{R}}$ iff $\neg A(a) \notin \mathcal{A}_{\mathsf{R}}$. ♢

Regarding Function (F2), first note that positive CQ literals in some χ_i with $i \in [1, k]$, cannot be contradicted by some $\mathcal{J}_j$ with $i \neq j$ if $\mathcal{J}_i$, the model of χ_i, and $\mathcal{J}_j$ both satisfy a common ABox type (e.g., if we regard an ontology $\mathcal{O} = \{A \sqsubseteq \neg B\}$ with $A \in \mathsf{N_{RC}}$ and assume that $\mathcal{J}_i$ satisfies a CQ $\exists y.B(a) \wedge A(y)$, $\mathcal{J}_j \models A(a)$ would be contradictory; but the latter cannot hold if the interpretations satisfy $\mathcal{O}$, and a common ABox type or, alternatively, share one domain and respect rigid names). Since the latter is ensured in our tests, it remains to consider the negative CQ literals occurring in some χ_i, $i \in [1, k]$. Observe that, apart from the agreement on the named individuals, we assume the domains of $\mathcal{J}_1, \dots, \mathcal{J}_k$ to be disjoint (on the unnamed part). Given $\mathsf{N_{RR}} = \emptyset$ and the connectedness of CQs, this leads to the fact that such literals can only be contradicted via rigid names by single elements which satisfy those names; and φ then must be *tree-shaped*[1].

Definition 5.3 (Tree-Shaped) A CQ φ is *tree-shaped* if it does not contain individual names and G_φ is a tree. If $\varphi \neq \emptyset$, then the tree has a unique root, the *root (variable)* of φ.

[1] The definition of $\mathrm{Con}(\varphi)$ is similar to the notion of "rolled-up" queries used by [Ros07].

For a tree-shaped CQ φ with root variable x, we set $\mathrm{Con}(\varphi) := \mathrm{Con}'(\varphi, x)$, where

$$\mathrm{Con}'(\varphi, y) := \bigsqcap_{A(y)\in\varphi} A \sqcap \bigsqcap_{R(y,z)\in\varphi} \exists R.\mathrm{Con}'(\varphi, z). \qquad \diamondsuit$$

In what follows, we assume that all concepts of the form $\mathrm{Con}(\varphi)$ for all tree-shaped CQs $\varphi \in \mathcal{Q}_\Phi$ also occur in $\mathcal{O}$.

In view of this definition, we can fully characterize the satisfaction of (tree-shaped) CQs in $\mathcal{Q}_\Phi$ based on rigid concept names in the anonymous part of an interpretation. Specifically, we use sets of rigid concept names as *witnesses*.

Definition 5.4 (Witness) A set $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ is a *witness* of a concept C w.r.t. $\mathcal{O}$ if $\bigsqcap \mathcal{B} \sqsubseteq C$ or if there are roles $R_1, \dots, R_\ell \in \mathsf{N}_{\mathsf{R}}$ and concepts $C_1, \dots, C_\ell \in \mathbb{S}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R_1.C_1$, $\mathcal{O} \models C_i \sqsubseteq \exists R_{i+1}.C_{i+1}$ for all $i \in [1, \ell-1]$, and $\mathcal{O} \models C_\ell \sqsubseteq C$. $\mathcal{B}$ is a *witness* of a tree-shaped CQ φ w.r.t. $\mathcal{O}$ if it is a witness of $\mathrm{Con}(\varphi)$ w.r.t. $\mathcal{O}$.

Let further $\mathcal{I}$ be the canonical interpretation for a knowledge base $\langle \mathcal{O}, \mathcal{A}\rangle$, where $\mathcal{A}$ is an arbitrary ABox. Then, $\mathcal{B}$ is a *witness* of an element $u_{\varrho R_1 C_1 \dots R_\ell C_\ell} \in \Delta^{\mathcal{I}}_{\mathsf{u}}$ w.r.t. $\langle \mathcal{O}, \mathcal{A}\rangle$ if $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R_1.C_1$, $\mathcal{O} \models C_i \sqsubseteq \exists R_{i+1}.C_{i+1}$ for all $i \in [1, \ell-1]$, and $u_\varrho \in (\bigsqcap \mathcal{B})^{\mathcal{I}}$ or $\varrho \in \mathsf{N}_{\mathsf{I}}(\mathcal{A}) \cap (\bigsqcap \mathcal{B})^{\mathcal{I}}$. The set of all witnesses of an unnamed element e w.r.t. $\mathcal{O}$ is denoted by $\mathcal{W}_{\mathcal{O}}(e)$. For all $e \in \Delta^{\mathcal{I}} \setminus \Delta^{\mathcal{I}}_{\mathsf{u}}$, we define $\mathcal{W}_{\mathcal{O}}(e) = \emptyset$. $\diamondsuit$

We sometimes say that an individual *satisfies* a witness $\mathcal{B}$ if it satisfies $\bigsqcap \mathcal{B}$. It is easy to see that, if a model of $\mathcal{O}$ contains an element that satisfies a witness of a CQ φ, then this model satisfies φ.

Lemma 5.5 *Let $\mathcal{I}$ be a model of $\mathcal{O}$ and $\mathcal{B}$ be a witness of a tree-shaped CQ φ w.r.t. $\mathcal{O}$. Then, $\mathcal{I} \models \exists x.\mathcal{B}(x)$ implies that $\mathcal{I} \models \varphi$.* $\square$

Based on the above observations, we collect CQs that may occur negated in some χ_i, $i \in [1, k]$, in the additional data to be able to ensure that none of their witnesses is satisfied in any of $\mathcal{J}_1, \dots, \mathcal{J}_k$. Observe that the additional data thus consists of a number of assertions and queries that is polynomial in the size of Φ. The conditions we then test are captured by a property of *r-completeness* similar to the one for $\mathcal{EL}$-LTL with global GCIs (see Definition 4.13).

To simplify the presentation of that property, we also represent the CQs to be satisfied as ABoxes. For all $i \in [1, k]$, let $Q_i := \{\varphi_j \mid p_j \in W_i\}$, and let $\mathcal{A}_{Q_i}$ denote the ABox obtained from Q_i by instantiating all variables x with fresh individual names a_x. We collect all these new individual names in the set $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$. Observe that, because of our assumption that the CQs in Φ have no variables in common, each $a_x \in \mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ can be unambiguously associated to a CQ containing x.

Definition 5.6 (r-complete) A tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg})$ consisting of an ABox type $\mathcal{A}_{\mathsf{R}}$ for $\mathcal{O}$ and a set $Q_{\mathsf{R}}^{\neg} \subseteq \mathcal{Q}_\Phi$ is *r-complete* (w.r.t. $\mathcal{W}$ and $\mathcal{O}$) if the following hold for all $i \in [1, k]$:

(C1) $\mathcal{K}^i_{\mathsf{R}} := \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_i}\rangle$ is consistent.

(C2) For all $p_j \in \overline{W_i}$, we have $\mathcal{K}^i_{\mathsf{R}} \not\models \varphi_j$.

(C3) For all tree-shaped CQs $\varphi \in Q_{\mathsf{R}}^{\neg}$ and all witnesses $\mathcal{B}$ of φ w.r.t. $\mathcal{O}$, we have $\mathcal{K}^i_{\mathsf{R}} \not\models \exists x.\mathcal{B}(x)$.

(C4) For all $W \in \mathcal{W}$ and $p_j \in \overline{W}$, we have $\varphi_j \in Q_{\mathsf{R}}^{\neg}$. ♢

The first two conditions together ensure that, for all considered worlds W_i with $i \in [1, k]$, exactly the queries specified by W_i *can* be satisfied w.r.t. $\mathcal{O}$ if the assertions in $\mathcal{A}_{\mathsf{R}}$ are taken into account. Condition (C3) additionally ensures that the queries induced by the propositions in $\overline{W}_i$ are not entailed based on the rigid names, by requiring that the canonical model of $\mathcal{K}_{\mathsf{R}}^i$ does not satisfy any of the witnesses of the tree-shaped queries in $Q_{\mathsf{R}}^{\neg}$ (see Lemma 2.13). In line with Conditions (C1)–(C3), the last condition makes sure that only queries from $Q_{\mathsf{R}}^{\neg}$ may be induced by elements of $\overline{W}$ for all $W \in \mathcal{W}$.

The existence of an r-complete tuple w.r.t. $\mathcal{W}$ fully characterizes the r-satisfiability of $\mathcal{W}$.

Lemma 5.7 *$\mathcal{W}$ is r-satisfiable w.r.t. $\langle \mathcal{O}, \emptyset \rangle$ iff there is an r-complete tuple w.r.t. $\mathcal{W}$ and $\mathcal{O}$.* □

Recall that the mapping ι is irrelevant in view of the TKB we consider by Fact 3.15. The proof of the lemma is split over the following two subsections. Subsequently, we describe how this lemma can be used to decide the entailment problem using only polynomial space.

If $\mathcal{W}$ is r-satisfiable, then there is an r-complete tuple w.r.t. $\mathcal{W}$ and $\mathcal{O}$

Given the r-satisfiability of $\mathcal{W}$, there are interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$ over a shared domain Δ as specified in Definition 3.12. We hence can define a tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg})$ as follows:

$$\begin{aligned}
\mathcal{A}_{\mathsf{R}} &:= \{A(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\Phi), A \in \mathsf{N}_{\mathsf{RC}}(\mathcal{O}), a^{\mathcal{J}_1} \in A^{\mathcal{J}_1}\} \cup \\
&\qquad \{\neg A(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\Phi), A \in \mathsf{N}_{\mathsf{RC}}(\mathcal{O}), a^{\mathcal{J}_1} \notin A^{\mathcal{J}_1}\}, \\
Q_{\mathsf{R}}^{\neg} &:= \{\varphi_j \in \mathcal{Q}_\Phi \mid p_j \notin \textstyle\bigcap \mathcal{W}\}.
\end{aligned}$$

The proof that $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg})$ is r-complete is straightforward, by regarding the conditions in Definition 5.6. According to Definition 5.2, $\mathcal{A}_{\mathsf{R}}$ is an ABox type for $\mathcal{O}$, and $Q_{\mathsf{R}}^{\neg} \subseteq \mathcal{Q}_\Phi$ holds obviously.

Condition (C1) is satisfied since each $\mathcal{J}_i$, $i \in [1, k]$, can be extended to a model $\mathcal{J}_i'$ of $\mathcal{K}_{\mathsf{R}}^i$ by appropriately defining the interpretations of the new individual names a_x that are introduced by $\mathcal{A}_{Q_i}$. More precisely, we have $\mathcal{J}_i \models \varphi_j$ for all $p_j \in W_i$ by assumption (see Definition 3.12); let π be the corresponding homomorphism. For each variable x occurring in φ_j, we can then copy the element $d := \pi(x) \in \Delta$, add the new individual e to the domain of $\mathcal{J}_i'$, set $a_x^{\mathcal{J}_i'} := e$, and interpret a_x as d is interpreted. Observe that we have to copy the element to overcome possible violations of the UNA, since a_x is an individual name. To see that the resulting interpretation is well-defined, we refer to the construction of $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ (see the part above Definition 5.6).

Condition (C2) is shown by contradiction. Assume that there are an $i \in [1, k]$ and a $p_j \in \overline{W_i}$ such that $\mathcal{K}_{\mathsf{R}}^i \models \varphi_j$. Given the proof for Condition (C1), we get $\mathcal{J}_i' \models \varphi_j$, which leads to $\mathcal{J}_i \models \varphi_j$ because φ_j does not contain any of the new individual names. But this contradicts the assumption that $\mathcal{J}_i \models \chi_i$.

Condition (C3) is also proven by contradiction. We assume that there are an $i \in [1, k]$, a tree-shaped CQ $\varphi_j \in Q_{\mathsf{R}}^{\neg}$, and a witness $\mathcal{B}$ of φ_j such that $\mathcal{K}_{\mathsf{R}}^i \models \exists x.\mathcal{B}(x)$ which, as

above, yields $\mathcal{J}_i \models \exists x.\mathcal{B}(x)$. However, by the above definition of $Q^{\neg}_{\mathsf{R}}$, there must be an $i' \in [1,k]$ such that $p_j \notin W_{i'}$, and thus $\mathcal{J}_{i'} \not\models \varphi_j$. Lemma 5.5 then yields that $\mathcal{J}_{i'} \not\models \exists x.\mathcal{B}(x)$ which, given $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$, contradicts the assumption that $\mathcal{J}_i$ and $\mathcal{J}_{i'}$ share one domain and respect the rigid names.

Condition (C4) is trivially satisfied.

If there is an r-complete tuple w.r.t. $\mathcal{W}$, then $\mathcal{W}$ is r-satisfiable.

The proof of the converse direction is more involved. We assume $(\mathcal{A}_{\mathsf{R}}, Q^{\neg}_{\mathsf{R}})$ to be the r-complete tuple given and, as before, follow the lines of Definition 3.12. The goal is thus to show that we can construct interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$ as required. Recall that we do not need to specifically define an interpretation $\mathcal{I}_0$ for time point 0 since, for any ι given, $\mathcal{J}_{\iota(0)}$ is a model of $\mathcal{A}_0 = \emptyset$ and $\chi_{\iota(0)}$ (see Fact 3.15). In a nutshell, we integrate the canonical interpretations of the KBs in Condition (C1) of Definition 5.6 to construct the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$.

In what follows, we first provide auxiliary definitions, then define $\mathcal{J}_1, \dots, \mathcal{J}_k$, and subsequently prove that the interpretations are as required. Note that, in the remainder of the proof, we generally do not reference Definition 5.6 explicitly if we refer to Conditions (C1)–(C4).

For all $i \in [1,k]$, consider the following definitions and observations.

- Let $\mathcal{I}_i := \mathcal{I}_{[\mathcal{K}^i_{\mathsf{R}}]^+}$ be the canonical interpretation of the KB $[\mathcal{K}^i_{\mathsf{R}}]^+$ obtained from $\mathcal{K}^i_{\mathsf{R}}$ by removing the negated assertions from $\mathcal{A}_{\mathsf{R}}$.

  We can establish the following fact in the way Fact 4.15 is obtained for $\mathcal{EL}$-LTL.

  Fact 5.8 *$\mathcal{I}_i$ is a model of $\mathcal{K}^i_{\mathsf{R}}$.*

- We define $\Delta^{\mathcal{I}_i}_{\mathsf{a}} := \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}} \cap \Delta^{\mathcal{I}_i}$ to distinguish the elements contained in $\mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$ and similarly write $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$ for the set containing the unnamed domain elements unique to the canonical interpretation $\mathcal{I}_i$. Moreover, we write e^i instead of e for the elements in these sets.

  Thus, the domain of $\mathcal{I}_i$ is composed of the pairwise disjoint sets $\mathsf{N}_{\mathsf{I}}(\Phi)$, $\Delta^{\mathcal{I}_i}_{\mathsf{a}}$, and $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$.

  Fact 5.9 *The set $\mathsf{N}_{\mathsf{I}}(\Phi)$, all sets $\Delta^{\mathcal{I}_i}_{\mathsf{a}}$ with $i \in [1,k]$, and all sets $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$ with $i \in [1,k]$ are pairwise disjoint.*

We next construct the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k$ as required for the r-satisfiability of $\mathcal{W}$; that is, all share one domain, they respect rigid names, and each $\mathcal{J}_i$ with $i \in [1,k]$ is a model of $\mathcal{O}$ and $\chi_i = \bigwedge_{p_j \in W_i} \varphi_j \wedge \bigwedge_{p_j \in \overline{W_i}} \neg\varphi_j$. To this end, we join the interpretations $\mathcal{I}_i$. The idea is that, for all $i \in [1,k]$, $\mathcal{I}_i$ represents the (flexible) parts specific to $\mathcal{J}_i$ and, for the interpretation of the rigid symbols in $\mathcal{J}_i$, all $\mathcal{I}_j$ with $j \in [1,k]$ are considered. Of course, the interpretation of the flexible symbols then cannot be solely based on $\mathcal{I}_i$ but has to be adjusted. In this way, we ensure that all of $\mathcal{J}_1, \dots, \mathcal{J}_k$ interpret the rigid concept names in the same way.

The common domain Δ is defined as follows:

$$\Delta := \mathsf{N_I}(\Phi) \cup \bigcup_{i=1}^{k} (\Delta_{\mathsf{a}}^{\mathcal{I}_i} \cup \Delta_{\mathsf{u}}^{\mathcal{I}_i}).$$

$\mathcal{J}_i$ is specified below, for $i \in [1, k]$:

- For all $a \in \mathsf{N_I}(\Phi)$: $a^{\mathcal{J}_i} := a$.
- For all rigid concept names A: $A^{\mathcal{J}_i} := \bigcup_{j=1}^{k} A^{\mathcal{I}_j}$.
- For all flexible concept names A:

$$A^{\mathcal{J}_i} := A^{\mathcal{I}_i} \cup \bigcup_{j=1}^{k} \bigcup_{\substack{\mathcal{B} \subseteq \mathsf{N_{RC}}(\mathcal{O}),\\ \mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq A}} \Big(\bigsqcap \mathcal{B}\Big)^{\mathcal{I}_j} \cup \bigcup_{j=1}^{k} \{e \in A^{\mathcal{I}_j} \cap \Delta_{\mathsf{u}}^{\mathcal{I}_j} \mid \mathcal{W}_{\mathcal{O}}(e) \neq \emptyset\}.$$

- For all (flexible) role names R:

$$R^{\mathcal{J}_i} := R^{\mathcal{I}_i} \cup \bigcup_{j=1}^{k} \{(d, e) \in R^{\mathcal{I}_j} \cap \left(\Delta^{\mathcal{I}_j} \times \Delta_{\mathsf{u}}^{\mathcal{I}_j}\right) \mid d \notin \mathsf{N_I}(\Phi), \mathcal{W}_{\mathcal{O}}(e) \neq \emptyset\}.$$

We thus have constructed interpretations $\mathcal{J}_1, \ldots, \mathcal{J}_k$ that share the same domain and respect the rigid concept names since, for all $A \in \mathsf{N_{RC}}$ and $i \in [1, k]$, the definition of $A^{\mathcal{J}_i}$ is independent of i. Most parts of the specification should be straightforward. Note that the cases referring to witnesses would not have to explicitly mention $\Delta_{\mathsf{u}}^{\mathcal{I}_j}$ (see Definition 5.4), but we include this information to ease both understanding and the proofs. For simplifying the proofs, we also exclude the named individuals in the very last case, though such tuples are obviously included in the interpretation in $\mathcal{J}_i$ because of $\mathsf{N_I}(\Phi) \subseteq \Delta^{\mathcal{I}_i}$, Fact 5.9, and the fact that $R^{\mathcal{I}_i} \subseteq R^{\mathcal{J}_i}$.

Before proving that $\mathcal{J}_1, \ldots, \mathcal{J}_k$ fulfill all requirements for the r-satisfiability of $\mathcal{W}$, we next establish some auxiliary results. Specifically, we show connections between $\mathcal{J}_1, \ldots, \mathcal{J}_k$ and the canonical interpretations, which help to clarify the picture of the former. First, we establish a basic connection between the interpretations $\mathcal{J}_i$ and $\mathcal{I}_i$ concerning the interpretation of role names.

Lemma 5.10 *For all $i \in [1, k]$, elements $d, e \in \Delta^{\mathcal{I}_i}$, and role names $R \in \mathsf{N_R}$, we have:*

$$(d, e) \in R^{\mathcal{J}_i} \text{ iff } (d, e) \in R^{\mathcal{I}_i}.$$

Proof. ($\Leftarrow$) This direction follows directly from the definition of $R^{\mathcal{J}_i}$.

($\Rightarrow$) The definition of $\mathcal{J}_i$ implies that there is a $j \in [1, k]$ such that $(d, e) \in R^{\mathcal{I}_j}$. We thus only have to regard the case that $j \neq i$ and can assume that e has a witness w.r.t. $\mathcal{O}$. Since $e \in \Delta^{\mathcal{I}_i}$, Fact 5.9 however implies that $e \in \mathsf{N_I}(\Phi)$. Hence, $j \neq i$ is impossible since named domain elements have no witnesses according to Definition 5.4. □

There is a similar connection between the interpretations of concepts in $\mathcal{J}_i$ and $\mathcal{I}_j$.

Lemma 5.11 *For all concepts $C \in \mathbb{S}(\mathcal{O})$ and $i, j \in [1, k]$, the following hold.*

a) *For all $e \in \Delta^{\mathcal{I}_i}$, we have $e \in C^{\mathcal{J}_i}$ iff $e \in C^{\mathcal{I}_i}$.*

b) *For all $e \in \Delta^{\mathcal{I}_j}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ such that $j \neq i$ and $\mathcal{W}_{\mathcal{O}}(e) = \emptyset$, we have $e \in C^{\mathcal{J}_i}$ iff there is a set $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq C$.*

c) *For all $e \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ such that $\mathcal{W}_{\mathcal{O}}(e) \neq \emptyset$, we have $e \in C^{\mathcal{J}_i}$ iff $e \in C^{\mathcal{I}_j}$.*

Proof. The items are proven simultaneously by induction over the structure of C. We start with the base cases.

- Let $C \in \mathsf{N}_{\mathsf{C}}$. ($\Leftarrow$) For all items, this direction is a direct consequence of the definition of $\mathcal{J}_i$. ($\Rightarrow$) We again consider the definition of $C^{\mathcal{J}_i}$.

  We first consider $e \in \mathsf{N}_{\mathsf{I}}(\Phi)$ to partially prove a). For $C \in \mathsf{N}_{\mathsf{RC}}$, the fact that each $\mathcal{I}_j$ with $j \in [1, k]$ is a model of the ABox type $\mathcal{A}_{\mathsf{R}}$ implies $e \in C^{\mathcal{I}_i}$. For flexible concept names C, Fact 5.9 yields two options: either $e \in C^{\mathcal{I}_i}$, or $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ for some $j \in [1, k]$ and $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq C$. Since both $\mathcal{I}_i$ and $\mathcal{I}_j$ are models of $\mathcal{A}_{\mathsf{R}}$, the latter yields $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_i}$ which, given $\mathcal{I}_i \models \mathcal{O}$, implies that $e \in C^{\mathcal{I}_i}$.

  Regarding the other elements e in a), b), and c), which means that $e \notin \mathsf{N}_{\mathsf{I}}(\Phi)$, Fact 5.9 implies for $C \in \mathsf{N}_{\mathsf{RC}}$ that $e \in C^{\mathcal{J}_i}$ iff $e \in C^{\mathcal{I}_j}$. For $C \in \mathsf{N}_{\mathsf{C}} \setminus \mathsf{N}_{\mathsf{RC}}$ and ($\Rightarrow$), we additionally have to remark that each $\mathcal{I}_j$ with $j \in [1, k]$ is a model of $\mathcal{O}$ to get $e \in C^{\mathcal{I}_j}$.

- Let $C = \top$. Both $e \in C^{\mathcal{J}_i}$ and $e \in C^{\mathcal{I}_i}$ ($e \in C^{\mathcal{I}_j}$) must hold by Definition 2.4, for a) (for b) and c)).

We continue with the induction step.

- Let $C = C_1 \sqcap C_2$. For a) (c)), the induction hypothesis directly yields the equivalence between $e \in C_1^{\mathcal{J}_i} \cap C_2^{\mathcal{J}_i}$ and $e \in C_1^{\mathcal{I}_i} \cap C_2^{\mathcal{I}_i}$ ($e \in C_1^{\mathcal{I}_j} \cap C_2^{\mathcal{I}_j}$).

  We consider b). ($\Rightarrow$) By the induction hypothesis $e \in (C_1 \sqcap C_2)^{\mathcal{J}_i}$ implies that there are sets $\mathcal{B}_1, \mathcal{B}_2 \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B}_1 \sqcap \bigsqcap \mathcal{B}_2)^{\mathcal{I}_j}$, $\mathcal{O} \models \bigsqcap \mathcal{B}_1 \sqsubseteq C_1$, and $\mathcal{O} \models \bigsqcap \mathcal{B}_2 \sqsubseteq C_2$. But then it also holds that $\mathcal{O} \models \bigsqcap \mathcal{B}_1 \sqcap \bigsqcap \mathcal{B}_2 \sqsubseteq C_1 \sqcap C_2$, and thus $e \in (C_1 \sqcap C_2)^{\mathcal{I}_j}$ because $\mathcal{I}_j \models \mathcal{O}$. ($\Leftarrow$) If $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq C_1 \sqcap C_2$ for some $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$, then we also have $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq C_1$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq C_2$. Together with the induction hypothesis, this leads to $e \in (C_1 \sqcap C_2)^{\mathcal{J}_i}$.

- Let $C = \exists R.C_1$. Regarding a), the definition of $R^{\mathcal{J}_i}$ yields that, for a given tuple $(e, d) \in R^{\mathcal{J}_i}$, we have $d \in \Delta^{\mathcal{I}_i}$. For such an element, the induction hypotheses implies that $d \in C_1^{\mathcal{J}_i}$ is equivalent to $d \in C_1^{\mathcal{I}_i}$. Lemma 5.10 additionally yields that $(e, d) \in R^{\mathcal{J}_i}$ is equivalent to $(e, d) \in R^{\mathcal{I}_i}$. The claim thus holds.

  Similarly, regarding c), the definition of $R^{\mathcal{J}_i}$ yields that, for a tuple $(e, d) \in R^{\mathcal{J}_i}$, we have $d \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ and, especially, $\mathcal{W}_{\mathcal{O}}(d) \neq \emptyset$. On the other hand, for a tuple $(e, d) \in R^{\mathcal{I}_j}$ where $e \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ and $\mathcal{W}_{\mathcal{O}}(e) \neq \emptyset$, Definitions 2.10 and 5.4 together

imply that also $d \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$ and $\mathcal{W}_{\mathcal{O}}(d) \neq \emptyset$. For both directions, we thus can focus on this type of successor and apply the induction hypotheses, which yields that $d \in C_1^{\mathcal{J}_i}$ is equivalent to $d \in C_1^{\mathcal{I}_j}$. Moreover, $\mathcal{W}_{\mathcal{O}}(d) \neq \emptyset$ implies that $(e, d) \in R^{\mathcal{J}_i}$ is equivalent to $(e, d) \in R^{\mathcal{I}_j}$. The claim thus also holds.

We consider b). ($\Rightarrow$) Since $i \neq j$, $e \in (\exists R.C_1)^{\mathcal{J}_i}$ by the definition of $R^{\mathcal{J}_i}$ implies that there is an element $u^j_{\varrho RD} \in C_1^{\mathcal{J}_i} \cap \Delta_{\mathsf{u}}^{\mathcal{I}_j}$ such that either $e = u^j_{\varrho}$ or $e = \varrho$ and $e \in \Delta_{\mathsf{a}}^{\mathcal{I}_j}$, and $\mathcal{W}_{\mathcal{O}}(u^j_{\varrho RD}) \neq \emptyset$. We can thus apply the induction hypotheses to obtain $u^j_{\varrho RD} \in C_1^{\mathcal{I}_j}$, and Lemma 2.15 then yields that $\mathcal{O} \models D \sqsubseteq C_1$. Since $\mathcal{W}_{\mathcal{O}}(e) = \emptyset$, Definition 5.4 additionally yields that there is a set $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R.D$, and thus $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R.C_1$.

($\Leftarrow$) If there is a set $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R.C_1$, then Definition 2.10 implies that there is an element $u^j_{\varrho RC_1} \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$ such that $(e, u^j_{\varrho RC_1}) \in R^{\mathcal{I}_j}$ and again either $e = u^j_{\varrho}$ or $e = \varrho$ and $e \in \Delta_{\mathsf{a}}^{\mathcal{I}_j}$. Definition 5.4 additionally yields that $\mathcal{W}_{\mathcal{O}}(u^j_{\varrho RC_1}) \neq \emptyset$. Thus, the definition of $\mathcal{J}_i$ yields $(e, u^j_{\varrho RC_1}) \in R^{\mathcal{J}_i}$. □

We finally show that $\mathcal{J}_i$ is in fact as intended.

Lemma 5.12 *For all $i \in [1, k]$, $\mathcal{J}_i$ is a model of $\mathcal{O}$.*

Proof. Consider a GCI $C \sqsubseteq D \in \mathcal{O}$ and an element $e \in C^{\mathcal{J}_i} \cap \Delta^{\mathcal{I}_j}$. According to Lemma 5.11, there are two options: (i) If a) or c) applies, then we have $e \in C^{\mathcal{I}_j}$. (ii) Otherwise, there is a set $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq C$. For both options, we then obtain $e \in D^{\mathcal{I}_j}$ since $C \sqsubseteq D \in \mathcal{O}$ and $\mathcal{I}_j \models \mathcal{O}$. This leads to $e \in D^{\mathcal{J}_i}$, again by Lemma 5.11. □

The next lemma provides the missing piece to our proof that $\mathcal{W}$ is r-satisfiable (see Definition 3.12) and shows that, for all $i \in [1, k]$, the interpretations $\mathcal{J}_i$ satisfy the corresponding conjunction χ_i of CQ literals. Regarding the positive literals, the proof is easy given that the ABox $\mathcal{A}_{Q_i}$ contains an instantiation of all these CQs, $\mathcal{A}_{Q_i}$ is satisfied by $\mathcal{I}_i$, and that $\mathcal{J}_i$ strongly depends on $\mathcal{I}_i$. The proof for the negative literals $\neg\varphi$ is based on the interpretation of roles in $\mathcal{J}_i$. In $\mathcal{J}_i$, domain elements of different interpretations $\mathcal{I}_j$ and $\mathcal{I}_\ell$ with $j, \ell \in [1, k]$ are related sparsely (i.e., at least one of two such elements must be a named individual). Given that we assume CQs to be connected, we specifically have that, if $\mathcal{J}_i$ satisfies φ, one must apply: either the corresponding homomorphism includes only domain elements of $\mathcal{I}_i$ and φ is satisfied in $\mathcal{I}_i$, or it contains no domain element of $\mathcal{I}_i$ (i.e., it especially contains no named individual), only elements of a single $\mathcal{I}_j$ with $j \in [1, k]$, and a witness of φ is satisfied in $\mathcal{I}_j$. Based on the assumed r-completeness of $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg})$, we therefore can show that $\mathcal{J}_i \models \neg\varphi$ by contradiction.

Lemma 5.13 *For all $i \in [1, k]$, $\mathcal{J}_i$ is a model of χ_i.*

Proof. We show that $\mathcal{J}_i$ is a model of every CQ literal in χ_i. Let φ first be a positive such literal. Since $\mathcal{A}_{Q_i}$ contains an instantiation of φ and $\mathcal{I}_i \models \mathcal{A}_{Q_i}$ by Condition (C1) and Fact 5.8[2], we know that there is a homomorphism π of φ into $\mathcal{I}_i$ that maps all

[2]Note that, in the remaining parts of the proof, we do not always explicitly refer to Fact 5.8 to justify the argument that $\mathcal{I}_i \models \mathcal{K}_{\mathsf{R}}^i$.

variables in φ to elements of $\Delta_{\mathsf{a}}^{\mathcal{I}_i}$; that is, π maps each such variable x to a_x. By the fact that $\Delta_{\mathsf{a}}^{\mathcal{I}_i} \subseteq \Delta$, since such elements do not have witnesses, and by Lemmas 5.10 and 5.11a) and b), π then is also a homomorphism of φ into $\mathcal{J}_i$.

Let now $\neg\varphi$ be a negative literal in χ_i. We proceed by contradiction and assume π to be a homomorphism of φ into $\mathcal{J}_i$. In particular, we can then assume that there is a single index $j \in [1, k]$ such that π maps all terms of φ to elements of $\Delta^{\mathcal{I}_j}$. To see this, note that φ is connected and that, for a role $R \in \mathsf{N}_\mathsf{R}$, a tuple $(d, e) \in R^{\mathcal{J}_i}$ exists only if the elements belong to the same domain $\Delta^{\mathcal{I}_j}$ by the definition of $R^{\mathcal{J}_i}$ and Fact 5.9.

If π maps all terms to elements of $\Delta^{\mathcal{I}_i}$, then Lemmas 5.10 and 5.11 yield that π is also a homomorphism of φ into $\mathcal{I}_i$. This contradicts the fact that $\mathcal{I}_i \models \neg\varphi$. The latter holds by Lemma 2.13 because the given r-complete tuple satisfies Condition (C2), which yields $\mathcal{K}_\mathsf{R}^i \not\models \varphi$ and hence $[\mathcal{K}_\mathsf{R}^i]^+ \not\models \varphi$; recall that $[\mathcal{K}_\mathsf{R}^i]^+$ is the KB for which $\mathcal{I}_i$ is the canonical interpretation.

Otherwise, we have $j \neq i$ and, given that $\mathsf{N}_\mathsf{I}(\Phi) \subseteq \Delta^{\mathcal{I}_i}$, can assume π to map at least one term to an element of $\Delta^{\mathcal{I}_j} \setminus \mathsf{N}_\mathsf{I}(\Phi)$. We can make two further observations about φ, due to the assumption that it is connected and the interpretation of roles in $\mathcal{J}_i$, which yields that elements from $\Delta^{\mathcal{I}_j} \setminus \mathsf{N}_\mathsf{I}(\Phi)$ can only be related to unnamed elements. First, no term of φ can be mapped to an element of $\mathsf{N}_\mathsf{I}(\Phi)$, which means that φ does not contain individual names (see Definition 2.7). Second, for all role atoms $R(y, z) \in \varphi$, we have $\mathcal{W}_\mathcal{O}(\pi(z)) \neq \emptyset$ and either (i) $\pi(y) \in \Delta_{\mathsf{a}}^{\mathcal{I}_j} \cup \Delta_{\mathsf{u}}^{\mathcal{I}_j}$ and $\mathcal{W}_\mathcal{O}(\pi(y)) = \emptyset$ or (ii) $\pi(y) \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$ and $\mathcal{W}_\mathcal{O}(\pi(y)) \neq \emptyset$. Hence, there is at most one variable in φ which π maps to an element e such that $\mathcal{W}_\mathcal{O}(e) = \emptyset$. Thus, φ contains only role connections starting from this variable, and role connections between two variables (mapped by π to elements having witnesses) exist only via a single role and in one direction. This means that φ is tree-shaped.

We now show that there is a witness $\mathcal{B}$ of φ w.r.t. $\mathcal{O}$ such that $(\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ is not empty. Let x be the root variable of φ. Then, Definition 5.3 and our assumption that π is a homomorphism of φ into $\mathcal{J}_i$, yield that $\pi(x) \in \mathrm{Con}(\varphi)^{\mathcal{J}_i}$.

- If $\mathcal{W}_\mathcal{O}(\pi(x)) = \emptyset$, then Lemma 5.11 yields that there is a witness $\mathcal{B}$ of $\mathrm{Con}(\varphi)$ w.r.t. $\mathcal{O}$ such that $\pi(x) \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ which is also a witness of φ by Definition 5.4.[3]
- If $\mathcal{W}_\mathcal{O}(\pi(x)) \neq \emptyset$, then $\pi(x)$ is of the form $u^j_{\varrho RC} \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$, and Lemma 5.11 yields that $\pi(x) \in \mathrm{Con}(\varphi)^{\mathcal{I}_j}$. From Lemma 2.15, we then get that $\mathcal{O} \models C \sqsubseteq \mathrm{Con}(\varphi)$. Thus, Definition 5.4 implies that the witness of $u^j_{\varrho RC}$, which is instantiated in $\mathcal{I}_j$, is also a witness of $\mathrm{Con}(\varphi)$ and hence of φ.

However, by Condition (C4) we know that $\varphi \in Q_\mathsf{R}^\sqsupset$. Hence, by Condition (C3), we have $[\mathcal{K}_\mathsf{R}^j]^+ \not\models \exists x.\mathcal{B}(x)$, and thus $\mathcal{I}_j \models \neg\exists x.\mathcal{B}(x)$ by Lemma 2.13. This contradicts the fact that $(\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ is not empty. □

This finishes the proof of Lemma 5.7.

[3] Recall that we assume that $\mathrm{Con}(\varphi)$ occurs in $\mathcal{O}$.

5.2 Combined Complexity

In this section, we establish the combined complexity of TCQ entailment, based on procedures that solve the satisfiability problem as outlined in Section 3.3. The results of the previous section reveal that the amount of information critical for testing the DL part of that problem is polynomial (in the size of Φ) if rigid role names are not considered. This allows us to integrate the LTL with the DL test in a nondeterministic algorithm using only polynomial space, based upon Algorithm 3.1. Regarding the remaining case with rigid role names, we however illustrate subsequently that the LTL and DL features may interact in a way that requires reasoning to consider an exponential amount of information. Specifically, we prove that entailment is co-NExpTime-complete.

5.2.1 With(out) Rigid Concept Names

In this section, we show that TCQ entailment is in PSpace w.r.t. combined complexity, which matches the hardness given by satisfiability in LTL. The key insight given by the previous section refers to the DL part of the TCQ satisfiability problem: the exponentially large set $\mathcal{W}$ does not have to be stored for testing the conditions characterizing its r-satisfiability if rigid roles are not present. This observation allows us to specify a nondeterministic procedure for deciding satisfiability—and thus also entailment—that only uses space of size polynomial in the input. The idea is described in Algorithm 3.1: we first guess an ABox type $\mathcal{A}_{\mathsf{R}}$ and a set $Q_{\mathsf{R}}^{\neg} \subseteq \mathcal{Q}_\Phi$ and then test the satisfiability of the LTL formula Φ^{pa} as it is done in Algorithm 2.1, but ensure additionally that the guessed worlds satisfy the conditions characterizing r-satisfiability. This approach is in line with Lemma 3.13; specifically, it integrates the r-satisfiability with the t-satisfiability test. Recall that we can disregard ι since its construction is trivial (i.e., we can use the first guessed world $W_i \in \mathcal{W}$ and define $\iota(0) = i$).

In our algorithm, we refer to the functions `KBCONSISTENT` and `CQNOTENTAILED` that decide knowledge base consistency and CQ non-entailment, respectively. We can assume `KBCONSISTENT` to run in polynomial time in the size of both the input KB [Bra04, Thm. 5] and the nondeterministic algorithm `CQNOTENTAILED` to run in polynomial time in the size of the input KB and given query [Ros07, Thm. 2]. Furthermore, we consider an enumerator `WITNESSENUM` which, given a CQ φ and an ontology $\mathcal{O}$, enumerates all witnesses for φ w.r.t. $\mathcal{O}$ (see Definition 5.4) as follows:[4]

- Construct a graph containing a node for each concept in $\mathbb{S}(\mathcal{O})$, and an edge labeled by a role R from C to D iff $\mathcal{O} \models C \sqsubseteq \exists R.D$.
- Mark all nodes $C = \mathrm{Con}(\varphi)$ with $\varphi \in \mathcal{Q}_\Phi$; recall that we assume all those concepts to be contained in $\mathcal{O}$.
- Enumerate all $\mathcal{B} \subseteq \mathsf{N}_{\mathsf{RC}}(\mathcal{O})$ and return those for which there are a role R and concept C such that $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R.C$ from which a marked concept D is reachable.

It is easy to see that this approach is both sound and complete. The graph can be constructed by quadratically many P-tests [Bra04, Thm. 5], the marking can be done in

[4] This idea is also implicitly used in the form of the reachability relation $\rightsquigarrow$ in [BBL05; KKS12].

polynomial time (see Definition 5.3), and the subsequent reachability test only requires polynomial space. The latter is due to the facts that there are polynomially many possibilities for the concept $\exists R.C$, which can be enumerated, in the last item and that the reachability problem is in NLogSpace (e.g., see [AB09, p. 74]).

Based on the above procedures, we define our algorithm as follows.

Definition 5.14 Given a TCQ Φ and TKB $\mathcal{K} = \langle \mathcal{O}, \emptyset \rangle$ satisfiability of Φ w.r.t. $\mathcal{K}$ can be decided by running Algorithm 3.1:

- `GUESSDATA`: It guesses and returns a tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg})$ consisting of an ABox type $\mathcal{A}_{\mathsf{R}}$ for $\mathcal{O}$ and a set $Q_{\mathsf{R}}^{\neg} \subseteq \mathcal{Q}_{\Phi}$.

- `TESTRSAT`: Given $\Phi, \mathcal{O}, \mathcal{A}_i, d$ (i.e., the tuple guessed), i, s, p, and W, it defines $\mathcal{A}_{Q_W} := \{\varphi_j \mid p_j \in W\}$ and $\mathcal{K}_{\mathsf{R}} := \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_W} \rangle$, and returns *true* iff the following conditions are satisfied:

  (C1) Check if `KBCONSISTENT`$(\mathcal{K}_{\mathsf{R}})$ returns *true*.

  (C2) For each $p_j \in \overline{W}$: Check if `CQNOTENTAILED`$(\varphi_j, [\mathcal{K}_{\mathsf{R}}]^+)$ returns *true*.

  (C3) For each tree-shaped CQ $\varphi \in Q_{\mathsf{R}}^{\neg}$ and $\mathcal{B}$ in `WITNESSENUM`$(\varphi, \mathcal{O})$:
  Check if `CQNOTENTAILED`$(\exists x.\mathcal{B}(x), \mathcal{K}_{\mathsf{R}})$ returns *true*.

  (C4) For each $p_j \in \overline{W}$: Check if $\varphi_j \in Q_{\mathsf{R}}^{\neg}$. ♢

Observe that we have $n = 0$, so that our additional requirement that $s > n$ in Algorithm 3.1 does not have any effects, compared to Algorithm 2.1. The next lemma summarizes the goal of our extensions.

Lemma 5.15 *The nondeterministic algorithm described in Definition 5.14 decides TCQ satisfiability in $\mathcal{EL}$ w.r.t. a TKB $\langle \mathcal{O}, \emptyset \rangle$ and uses only polynomial space (in the size of all the input) if $\mathsf{N}_{\mathsf{RR}} = \emptyset$.*

Proof. For proving correctness, we consider the conditions in Lemma 3.13.

- Let the set $\mathcal{W}$ be defined as the set of all worlds W encountered during a run of the procedure. The mapping ι and the conditions referring to it can be ignored by Fact 3.15.

- Regarding t-satisfiability (see Definition 3.11), it is easy to see that the above definition of $\mathcal{W}$ fulfills the first condition. Since Algorithm 2.1 is correct by Lemma 2.21, we run it with Φ^{pa} as input, and our extensions neither change this algorithm nor do they ever return true on their own (i.e., independently of Algorithm 2.1), it is sound w.r.t. the second condition as well; and it is complete if our extensions never return false if an r-satisfiable set $\mathcal{W}$ exists.

- It thus remains to show that $\mathcal{W}$ is r-satisfiable iff the extensions do *not* return false on their own. By Lemma 5.7, we can consider Conditions (C1)–(C4) from Definition 5.6. For most of the cases, it is easy to see that the checks in Definition 5.14 are equivalent to the conditions. We only consider the second test in more detail since it does not directly correspond to Condition (C2). However, we can show that $\mathcal{K}_{\mathsf{R}} \not\models \varphi_j$ iff $[\mathcal{K}_{\mathsf{R}}]^+ \not\models \varphi_j$: ($\Leftarrow$) Given $[\mathcal{K}_{\mathsf{R}}]^+ \not\models \varphi_j$, Lemma 2.13 leads

to $\mathcal{I}_{[\mathcal{K}_\mathsf{R}]^+} \not\models \varphi_j$ and Fact 5.8 to $[\mathcal{K}_\mathsf{R}] \not\models \varphi_j$. ($\Rightarrow$) Every model of $\mathcal{K}_\mathsf{R}$ is also a model of $[\mathcal{K}_\mathsf{R}]^+$, especially the one that does not satisfy φ_j. Hence, it suffices to check the non-entailment $[\mathcal{K}_\mathsf{R}]^+ \not\models \varphi_j$.

We analyze the complexity. The nondeterministic guessing of the polynomially large sets $\mathcal{A}_\mathsf{R}$ and $Q^\neg_\mathsf{R}$ can be clearly done using only polynomial space. Regarding the parts of Algorithm 2.1, we refer to Lemma 2.21. Given the observation that we only adapt that algorithm w.r.t. the r-satisfiability testing, it only remains to consider the corresponding tests. The above descriptions of the applied subprocedures show that running all of them requires only polynomial space w.r.t. combined complexity; we apply them only for inputs of size polynomial in Φ and $\mathcal{K}$. Since all other checks described for testing the conditions can be done in polynomial time, our nondeterministic algorithm altogether uses only polynomial space. □

We thus can conclude this section with a positive result, which holds w.r.t. arbitrary TKBs by Lemma 3.7 and in the presence of rigid concept names. Further, note that the nondeterminism is not relevant regarding PSpace complexity according to the well-known result of Savitch [Sav70, Thm. 1].

Corollary 5.16 *TCQ entailment in $\mathcal{EL}$ is in* PSpace *regarding combined complexity if $\mathsf{N}_{\mathsf{RR}} = \emptyset$, even if $\mathsf{N}_{\mathsf{RC}} \neq \emptyset$.*

Altogether, we have shown that important and powerful features of LTL, (i) the possibility to discern exponentially many different time points and (ii) the nondeterminism provided by operators such as disjunction or negation, do not interact with the $\mathcal{EL}$ ontology in a way that is critical for the combined complexity of TCQ entailment, even if rigid concept names are considered. As it is the case for LTL, reasoning can still be done with a PSpace Turing machine.

5.2.2 With Rigid Role Names

In this section, we show that rigid role names may cause dangerous interactions between the LTL and the $\mathcal{EL}$ part that cannot be captured by PSpace Turing machines any more: the LTL features can discern exponentially many time points and nondeterministically choose specific assertions at each of them, and the $\mathcal{EL}$ part can correspondingly discern exponentially many (rigid) concepts instantiated by different individuals that are related invariant to time. For that reason, the LTL choices can be transferred selectively (i.e., addressing only some of the individuals) along these relations via the ontology and "saved" at those individuals—again via rigid names. We in the following show that such complex interactions lead to NExpTime-hardness of TCQ satisfiability; the proof is similar to the one for $\mathcal{EL}$-LTL for the case with rigid concept names in Section 4.8. The corresponding containment result directly follows from Lemmas 3.17 and 5.1.

Corollary 5.17 *TCQ entailment in $\mathcal{EL}$ is in* co-NExpTime *in combined complexity, even if $\mathsf{N}_{\mathsf{RR}} \neq \emptyset$.*

We prove co-NExpTime-hardness of entailment in the presence of rigid role names by reducing the 2^{n+1}-bounded domino problem to TCQ satisfiability. The idea is based on the features outlined above. The exponentially many different time points, each

associated with a specific rigid concept and individual instantiating it, represent the positions in the plane of the domino. We ensure that these individuals are, by a rigid role R, related to a common successor, a named individual used for synchronization. To tile the plane, we represent the domino types as flexible concepts and enforce the named individual to always satisfy one of them, by nondeterministically choosing the corresponding assertion. The ontology is used to transfer that choice to the R-predecessor that represents the position corresponding to the current time point and to save it at that individual via a rigid concept. In this way, we ensure that all positions and the chosen types are instantiated in *every world*, which allows us to enforce the matching conditions.

The basic idea of the reduction is thus the same as in the NExpTime-hardness proof for satisfiability in $\mathcal{EL}$-LTL regarding rigid concept names; note that the latter, in turn, is based on a reduction in [BGL12]. In those proofs, the synchronization of the domain individuals with the named individual is achieved via local GCIs and disjunction in the GCIs, respectively, allowed in the respective context. In contrast, we here need the rigid role to synchronize (a large enough subset of) the domain individuals with the named individual, which is done by using the latter as a common role successor of the former. Note that we cannot simply used global GCIs of the form $\top \sqsubseteq \dots$ instead since, then, all individuals would have to always satisfy all domino types.

Theorem 5.18 *TCQ entailment in $\mathcal{EL}$ is* co-NExpTime*-hard in combinded complexity if $\mathsf{N_{RR}} \neq \emptyset$.*

Proof. For the proof, we reduce the 2^{n+1}-bounded domino problem, for a domino system $\mathcal{D}$ with initial condition I, to checking the satisfiability of a TCQ $\Phi_{\mathcal{D},I}$ w.r.t. an $\mathcal{EL}$ TKB $\langle \mathcal{O}_{\mathcal{D},I}, \emptyset \rangle$, both containing rigid role names. Since the reduction of the domino system is very similar to the above mentioned reduction to satisfiability in $\mathcal{EL}$-LTL (see Theorem 4.8), we only point out the differences to that proof. In particular, we here apply exactly the same symbols and consider only one rigid role R in addition.

The rigid role is necessary here to transfer specific flexible concepts that are chosen nondeterministically through assertions on a named individual a to the entire domain; we cannot use local GCIs as allowed in $\mathcal{EL}$-LTL. However, as outlined above, it is actually enough to transfer the concepts to all R-predecessors of a if we require them to correspondingly represent the domino plane; in the proof for $\mathcal{EL}$-LTL, the plane is represented by arbitrary elements. In particular, the (flexible) *global*[5] concepts, whose interpretation is either empty or $\top$ and which are applied for synchronization in the other proof, have a slightly different interpretation here: their value is shared by a and all its R-predecessors.

Observe that we focus on the first part of the reduction and thus on an $\mathcal{EL}_\bot$-LTL formula; recall that $\bot$ is only eliminated in the very last step of the proof. We next regard all conjuncts of the target formula created in the proof for $\mathcal{EL}_\bot$-LTL (i.e., these conjuncts are $\mathcal{EL}_\bot$-LTL formulas) and in the following specify corresponding conjuncts (i.e., TCQs) and GCIs, such that $\Phi_{\mathcal{D},I}$ consists of a conjunction of these TCQs and $\mathcal{O}_{\mathcal{D},I}$ of the GCIs:

[5] Not to be confused with *rigid* or *always* (in time).

- All global GCIs (i.e., conjuncts that are GCIs prefixed by $\Box_F$) that do not contain $\bot$ can be directly considered as GCIs in $\mathcal{O}_{\mathcal{D},I}$.
- All $\mathcal{EL}_\bot$-LTL formulas that contain no GCIs (and hence are $\mathcal{EL}$ assertions combined by LTL operators) can be directly considered as TCQs and conjuncts of $\Phi_{\mathcal{D},I}$.
- To represent the exponentially many positions, an auxiliary concept $N \in \mathsf{N_C}$ is required to be instantiated by an R-predecessor of a in every world:

$$\Box \exists x.R(x,a) \wedge N(x).$$

- To express that a concept C is *global* (e.g., the three counters and their complements), which means that it is shared by a and all its R-predecessors, we can use an $\mathcal{EL}$ GCI: $\exists R.C \sqsubseteq C$. However, different from the other proof, where concepts C and their complements $\overline{C}$ are modeled using mutually exclusive assertions on a, and globality of the two concepts then automatically means that no element satisfies both of these concepts, we here have to explicitly require the latter for all the predecessors, because globality "only" expresses the transfer of concepts from a to its R-predecessors:

$$\neg \exists x.C(x) \wedge \overline{C}(x).$$

- The fact that every world is associated to exactly one global domino type $G_t \in \mathsf{N_C}$, which represents the chosen domino type t for the position the (current) world corresponds to, can be expressed similarly to the global concepts by considering the GCI $\exists R.G_t \sqsubseteq G_t$ for all $t \in T$ and the following TCQ:

$$\Box \bigvee_{t \in T} \Big(G_t(a) \wedge \bigwedge_{t' \in T \setminus \{t\}} \neg \exists x.G_{t'}(x) \Big)$$

  where T is the set of domino types in $\mathcal{D}$. The choices for the dominos of the neighbor positions can be modeled analogously.

The polynomial size of the TCQ $\Phi_{\mathcal{D},I}$ and ontology $\mathcal{O}_{\mathcal{D},I}$ then directly follows from the size of the $\mathcal{EL}_\bot$-LTL formula in the other reduction and the described modifications. The correctness of the reduction can readily be checked similarly. □

5.3 Data Complexity

Regarding data complexity, we show that TCQ entailment in $\mathcal{EL}$ is in P—the best result possible since CQ entailment is P-hard [Cal+06, Thm. 7]—if rigid symbols are not considered. Subsequently, we however show that the possibility of having assertions in arbitrary ABoxes in the TKB (and a fixed TCQ) leads to co-NP-hardness in the presence of rigid names. Nevertheless, containment in co-NP is given also for the case with rigid role names, which is shown at the end of the section.

5.3.1 Without Rigid Names

Also w.r.t. data complexity, deciding satisfiability and entailment is special if rigid symbols are disregarded, because the knowledge (to be) considered w.r.t. different time points is independent of each other. Regarding Lemma 3.13, observe that all possible sets $\mathcal{W}$, which are independent of the data, can be enumerated in time constant in the data. While the exponential number of possible mappings ι however represents an obstacle, at first glance, both the t-satisfiability and the r-satisfiability test themselves are not complex. The idea is therefore to integrate all the mappings that pass the r-satisfiability test while testing (i.e., without constructing them one after the other) and to consider this integrated representation, which is of polynomial size, in the t-satisfiability test. Altogether, this can be done in polynomial time in the data.

Theorem 5.19 *TCQ entailment in $\mathcal{EL}$ is in* P *in data complexity if* $\mathsf{N_{RC}} = \mathsf{N_{RR}} = \emptyset$.

Proof. We follow the basic approach of Lemma 3.13 and regard the satisfiability problem.

- To check the r-satisfiability of a set $\mathcal{W} = \{W_1, \dots, W_k\} \subseteq 2^{\{p_1,\dots,p_m\}}$ as proposed in Lemma 3.14, it suffices to check satisfiability of the conjunctions $\chi_i^{(i)}$ with $i \in [1,k]$ w.r.t. $\langle \mathcal{O}_i, \emptyset \rangle$ ($\chi_{\iota(i)}^{(k+i+1)}$ w.r.t. $\langle \mathcal{O}_{k+i+1}, \mathcal{A}_i \rangle$, where $i \in [0,n]$) individually by Lemma 3.16, since the conjunctions χ_i do not share concept or role names.

  In particular, we can then define $\mathcal{W}$ as the set of *all* sets W_i with $i \in [1,k]$ for which χ_i is satisfiable w.r.t. $\mathcal{O}$. This can be done in constant time w.r.t. the size of the input ABoxes.

  The mapping ι cannot be constructed in polynomial time, and neither be obtained by (nondeterministically) guessing one possible ι or enumerating the exponentially many possible mappings. For that reason, we first check for each $W_j \in \mathcal{W}$ and input ABox $\mathcal{A}_i$ whether χ_j is satisfiable w.r.t. $\langle \mathcal{O}, \mathcal{A}_i \rangle$. We collect all indices j that pass this test into the set $\iota'(i)$. In this way, we obtain all possible worlds, for each of the input ABoxes. Each of the conjunctions χ_j is of constant size and the number $|\mathcal{W}|$ of conjunctions to be considered per ABox is also constant. By Lemma 5.1, these tests can thus be done in polynomial time in the size of the input ABoxes. Each set $\iota'(i)$ is of constant size.

  Note that r-satisfiability is thus given by the definition of $\mathcal{W}$ and ι', which means that such sets with $|\mathcal{W}| \geq 1$ and $|\iota'(i)| \geq 1$ for all $i \in [0,n]$ exist.

- Lastly, t-satisfiability (see Definition 3.11) can be decided by an automaton similar to the one described in the proof of [BBL15b, Lem. 4.12] in P w.r.t. data complexity. Since $\mathcal{W}$ is defined to be maximal above, we can also use it for testing t-satisfiability. To be able to apply $\iota'(i)$ (i.e., instead of only a single possible mapping ι), we adapt the condition "$w_i = W_{\iota(i)}$" in Definition 3.11 to "$w_i = W_j$ for some $j \in \iota'(i)$". The result remains valid since the automaton used in the proof of that result in [BBL15b] can be adapted to check whether the first $n+1$ encountered worlds fall into the pre-specified sets of (constantly many) worlds $\iota'(i)$, instead of equality with a single pre-specified world $\iota(i)$.

This approach is sound since, if both the above polynomial-time tests succeed, then we can simply choose one ι among the many possible identified by ι' to fulfill the conditions of Lemma 3.13. And it is complete since the existence of some $\mathcal{W}$ and ι implies that the above defined set $\mathcal{W}$ is not empty and that we have $\iota(i) \in \iota'(i)$ for every $i \in [0, n]$. Hence, all the above tests succeed. This proves that our deterministic definitions of the maximal possible $\mathcal{W}$ and ι' suffice to satisfy Lemma 3.13, which means that we can decide TCQ satisfiability (and entailment) in P. □

5.3.2 With Rigid Names

If rigid symbols are considered, then assertions occurring in arbitrary ABoxes may constrain the satisfaction of CQs from Φ in models of the ontology, by implying rigid knowledge. We show that, although both the TCQ and the ontology are fixed regarding data complexity, this nondeterminism has to be taken into account on LTL level (e.g., an approach as in the previous section, where the mapping ι' is defined separately for each time point, does not work any more) and leads to co-NP-hardness of TCQ entailment. The corresponding containment result directly follows from Lemmas 3.17 and 5.1. That is, the possibility of nondeterministic guessing allows to decide satisfiability in polynomial time w.r.t. data complexity, even in the presence of rigid symbols in general. This is mainly due to the fact that the TCQ is considered as fixed, which means that the generally exponentially many possible worlds on the LTL level are not critical for time complexity.

Corollary 5.20 *TCQ entailment in $\mathcal{EL}$ is in* co-NP *w.r.t. data complexity, even if* $\mathsf{N}_{\mathsf{RR}} \neq \emptyset$.

We prove co-NP-hardness of entailment in the presence of rigid concept names by reducing the 3-SAT problem to TCQ satisfiability. Specifically, we consider one named individual c and one named individual for the positive and, respectively, negative form of each variable. The idea is to describe each clause in three consecutive ABoxes, one for each literal, in each of which the corresponding individual is related to c via a role S. The TCQ is used to enforce c to satisfy a concept T at one of these three time points—intuitively, the corresponding literal is selected to be true—and the ontology to "save" this selection using the GCI $\exists S.T \sqsubseteq A$ at the individual representing the literal via a rigid concept name A—representing the variable assignment. In the TCQ we then additionally ensure that the two individuals representing a variable and its negation, respectively, never satisfy A at the same time. An example is illustrated in Figure 5.1.

Theorem 5.21 *TCQ entailment in $\mathcal{EL}$ is* co-NP*-hard in data complexity if* $\mathsf{N}_{\mathsf{RC}} \neq \emptyset$, *even if* $\mathsf{N}_{\mathsf{RR}} = \emptyset$.

Proof. We show NP-hardness of TCQ satisfiability by reduction of the 3-SAT problem, which is known to be NP-complete [Kar72, Main Thm.]. We assume a propositional 3-CNF formula

$$\varphi = \bigwedge_{0 \leq i < \ell} l_{i,1} \vee l_{i,2} \vee l_{i,3}$$

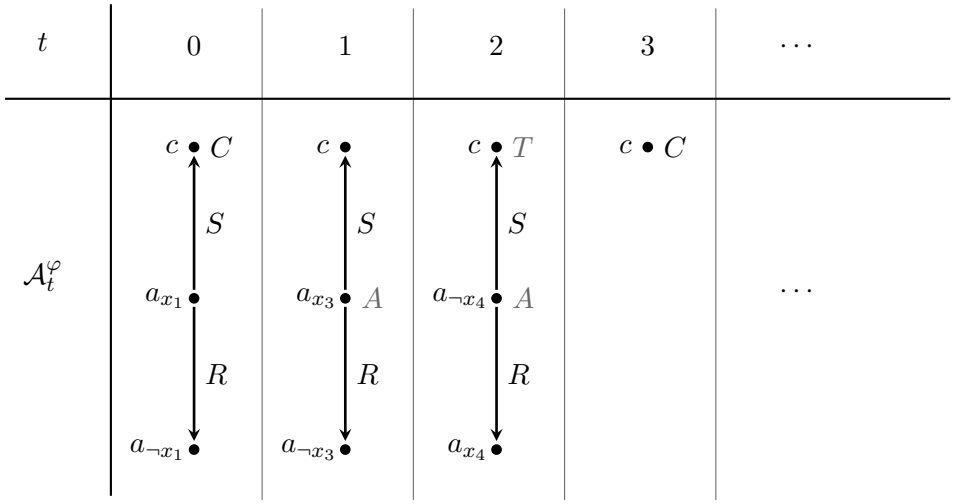

Figure 5.1: The content of the ABoxes encoding a 3-CNF formula $(x_1 \vee x_3 \vee \neg x_4) \wedge \ldots$; names in gray describe a possible extension to a model of Φ w.r.t. $\mathcal{K}_\varphi$.

with $\ell \geq 1$ to be given, $x_1, \ldots, x_m$ to be all the propositional variables occurring in φ, Lit_φ to be the set of all literals over these variables, and $\neg l$ to denote the complement of a literal l.

We construct a TCQ Φ and a TKB $\mathcal{K}_\varphi = \langle \mathcal{O}, (\mathcal{A}_t^\varphi)_{0 \leq t < 3\ell} \rangle$ such that φ is satisfiable iff Φ is satisfiable w.r.t. $\mathcal{K}_\varphi$. For that, we use the following symbols:

- an auxiliary individual name c to represent the clauses,
- individual names a_l for all literals $l \in \mathrm{Lit}_\varphi$,
- a rigid concept name A to represent a truth value assignment by marking true literals,
- a flexible concept name C to signal at t the beginning of the encoding of clause $\frac{t}{3}$,
- a flexible concept name T to identify which literal of a clause is satisfied,
- a role name S to relate a clause with its literals,
- a role name R to link each a_l to $a_{\neg l}$ to ensure that the truth assignment is consistent.

We use three ABoxes to model each clause of φ: one to represent the beginning of the encoding of a new clause and the first literal, and the following two for the other literals. The ABoxes $\mathcal{A}_t^\varphi$ with $t \in [0, 3\ell - 1]$ are defined as follows, for all $i \in [0, \ell - 1]$ and $j \in [1, 3]$:

$$\mathcal{A}_{3i}^\varphi := \{C(c)\},$$
$$\mathcal{A}_{3i+j-1}^\varphi := \{R(a_{l_{i,j}}, a_{\neg l_{i,j}}), S(a_{l_{i,j}}, c)\}.$$

Note that the size of all ABoxes $\mathcal{A}_j^\varphi$ together is linear in the size of φ, in line with the data complexity assumptions. Also in accordance with the latter, Φ and $\mathcal{O}$ are defined independently of the concrete input problem φ:

$$\begin{aligned}\Phi &:= \Box\Big((C(c) \to (T(c) \vee \bigcirc_F T(c) \vee \bigcirc_F \bigcirc_F T(c))) \\ &\qquad \wedge \neg\exists x, y.R(x,y) \wedge A(x) \wedge A(y)\Big), \\ \mathcal{O} &:= \{\exists S.T \sqsubseteq A\}.\end{aligned}$$

The TCQ Φ requires that, whenever $C(c)$ is satisfied, then this world or one of the two consecutive ones does satisfy $T(c)$. Intuitively, this represents a truth assignment for the literal l associated with that moment, and the ontology is used to save that assignment at the individual a_l via A, such that every world satisfies $A(a_l)$. Additionally, Φ ensures that individuals linked by R cannot both satisfy the rigid concept name A at the same time. Note that our reduction requires several features: quantified existential restriction, a rigid concept, and the Boolean negation and the temporal operators in the TCQ. In what follows, we show that there is an assignment $v\colon \{x_1, \dots, x_m\} \to \{0,1\}$ that satisfies φ iff Φ is satisfiable w.r.t. $\mathcal{K}_\varphi$.

($\Rightarrow$) Let v be such an assignment. We define the model $\mathfrak{I} = (\mathcal{I}_t)_{t\geq 0}$ of Φ w.r.t. $\mathcal{K}_\varphi$ with domain $\Delta := \{c, a_{x_1}, \dots, a_{x_m}, a_{\neg x_1}, \dots, a_{\neg x_m}\}$, where all individual names occuring in the ABoxes are interpreted as themselves as follows:

$$\begin{aligned}A^{\mathcal{I}_t} &:= \{a_l \mid l \in \mathsf{Lit},\ v(l) = 1\}, \\ T^{\mathcal{I}_t} &:= \{c \mid 0 \leq i < \ell,\ 1 \leq j \leq 3,\ t = 3i + j - 1,\ v(l_{i,j}) = 1\}, \\ C^{\mathcal{I}_t} &:= \{d \mid t < 3\ell,\ C(d) \in \mathcal{A}_t^\psi\}, \\ R^{\mathcal{I}_t} &:= \{(d,e) \mid t < 3\ell,\ R(d,e) \in \mathcal{A}_t^\varphi\}, \\ S^{\mathcal{I}_t} &:= \{(d,e) \mid t < 3\ell,\ S(d,e) \in \mathcal{A}_t^\varphi\}.\end{aligned}$$

We obviously have $\mathcal{I}_t \models \mathcal{A}_t^\varphi$ for all $0 \leq t < 3\ell$. Consider now the only GCI $\exists S.T \sqsubseteq A \in \mathcal{O}$. By the definition of $S^{\mathcal{I}_t}$ based on the ABox $\mathcal{A}_t^\varphi$, the left-hand side concept can only be satisfied by an individual of the form a_l. If $a_l \in (\exists S.T)^{\mathcal{I}_t}$, then we have $l = l_{i,j}$ for $t = i + 3j - 1$ and $c \in T^{\mathcal{I}_t}$ by the definition of $T^{\mathcal{I}_t}$; and also $v(l) = 1$. But then we also have $a_l \in A^{\mathcal{I}_t}$, which shows that $\mathfrak{I}$ is a model of $\mathcal{K}_\varphi$.

Since v satisfies each clause of φ, it is clear that $\mathfrak{I}$ satisfies the following implication at every time point by its definition, given the ABox definitions based on φ:

$$C(c) \to (T(c) \vee \bigcirc_F T(c) \vee \bigcirc_F \bigcirc_F T(c)).$$

Moreover, whenever $(d,e) \in R^{\mathcal{I}_t}$, then we must have $d = a_l$ and $e = a_{\neg l}$ for some $l \in \mathsf{Lit}$. By the definition of $A^{\mathcal{I}_t}$, we then cannot have both $d \in A^{\mathcal{I}_t}$ and $e \in A^{\mathcal{I}_t}$. This shows that $\mathfrak{I}$ satisfies the entire TCQ Φ.

($\Leftarrow$) Let $\mathfrak{I} = (\mathcal{I}_t)_{t\geq 0}$ be a model of Φ w.r.t. $\mathcal{K}_\varphi$ that interprets all individual names as themselves. We define $v(x_k) := 1$ if $a_{x_k} \in A^{\mathcal{I}_0}$, and $v(x_k) := 0$ otherwise.

Consider now an arbitrary clause $l_{i,1} \vee l_{i,2} \vee l_{i,3}$ of φ. Since $C(c) \in \mathcal{A}_{3i}^\varphi$, $\mathfrak{I} \models \Phi$ implies that there is an index $j \in [1,3]$ such that $c \in T^{\mathcal{I}_{3i+j-1}}$. By the definition of $\mathcal{A}_{3i+j-1}^\varphi$,

we also have $(a_{l_{i,j}}, c) \in S^{\mathcal{I}_{3i+j-1}}$, and thus $a_{l_{i,j}} \in A^{\mathcal{I}_{3i+j-1}} = A^{\mathcal{I}_0}$ follows from the GCI $\exists S.T \sqsubseteq A$.

If $l_{i,j}$ is a variable, then we directly get $v(l_{i,j}) = 1$ by the definition of v, which shows that the clause is satisfied by v. Otherwise, we have $l_{i,j} = \neg x_k$ for some $k \in [1, m]$. By the definition of $\mathcal{A}^{\varphi}_{3i+j-1}$, we know that $(a_{x_k}, a_{\neg x_k}) \in R^{\mathcal{I}_{3i+j-1}}$. Since $\mathfrak{I}$ satisfies Φ and $a_{\neg x_k} \in A^{\mathcal{I}_0}$, it cannot be the case that $a_{x_k} \in A^{\mathcal{I}_0}$. This means that $v(x_k) = 0$, and thus we also get $v(l_{i,j}) = 1$. □

We have thus completely classified TCQ entailment in $\mathcal{EL}$ regarding different settings with rigid symbols. In summary, we have shown that the main features of $\mathcal{EL}$, conjunction and qualified existential restriction, do not lead to critical interaction with LTL if rigid symbols are disregarded. Otherwise, this only holds for the case with rigid concept names under combined complexity assumptions. $\mathcal{EL}$ TKBs and LTL in queries can thus, respectively, be used with LTL satisfiability checking and CQ answering in $\mathcal{EL}$ "for free", if the focus is on data complexity. Regarding the latter, we have further proven co-NP-completeness if rigid symbols are included and thus that tractability is lost. Rigid roles lead also to a considerable increase in combined complexity, to co-NExpTime-completeness. As we show in the next chapter, which focuses on other Horn DLs, both these critical increases are due to the qualified existential restriction of $\mathcal{EL}$.

6 Temporal Query Entailment in $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$

In this chapter, we regard a Boolean TCQ Φ and a TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ in $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$ and investigate the combined and data complexity of the TCQ entailment problem. We specifically show that the former equals that of satisfiability in LTL and that the latter—ALogTime-completeness—only slightly increases compared to the baseline, the data complexity of CQ answering, which is in AC^0. In particular, these results are tight for all logics between $\mathit{DL\text{-}Lite}_{core}$ and $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$, even if rigid symbols are considered. Note that the complexities and the fact that they do not depend on the kind of rigid symbols considered are different from what we have shown for $\mathcal{EL}$ in Chapter 5. Although we here use ideas similarly applied in the previous chapter, the general consideration of rigid symbols, the features allowed in *DL-Lite*, and the complexities we target require us to considerably extend these approaches.

The main expressive features of $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$ are inverse roles, conjunction, and role inclusions (see the examples in Section 1.2). Different from $\mathcal{EL}$, *DL-Lite* ontologies only allow for unqualified existential restrictions. This often makes reasoning less complex, especially w.r.t. data complexity—as it is reflected in our results. Nevertheless, qualified existential restrictions as $\exists \mathtt{HasState.SwitchedOff}$ on the right-hand side of CIs can be simulated by using a concept $\exists \mathtt{HasOffState}$[1] instead and axioms as follows:

$$\mathtt{HasOffState} \sqsubseteq \mathtt{HasState}$$
$$\exists \mathtt{HasOffState}^{-} \sqsubseteq \mathtt{SwitchedOff}$$

Also in this chapter, our results are obtained by investigating procedures based on Lemma 3.13 solving TCQ satisfiability and focusing on satisfiability problems in LTL and DLs. The challenge thereby is to construct a corresponding set $\mathcal{W}$ of LTL worlds and mapping ι within specific time and space constraints. Recall that, regarding the sublinear data complexity we target here, not only the size of this exponentially large set but also the linear size of the mapping represents an obstacle.

In Section 6.1, we first propose a new characterization of r-satisfiability—the DL part of the TCQ satisfiability problem—in the presence of rigid names that is tailored to TCQs and $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$ and, different from existing characterizations (see Section 3.2), does not consider the set as $\mathcal{W}$ a whole. We then apply this approach in Section 6.2 on combined complexity for obtaining the PSpace result. Subsequently, we refine it in Section 6.3 in a way that allows for defining FO rewritings of r-satisfiability. Based on those, the data complexity result is obtained in Section 6.4.

Throughout the chapter, we use the notation of Section 3.2. For ease of presentation, we furthermore allow basic concepts to occur in ABoxes in TKBs, without loss of generality.

[1] Recall that unqualified existential restrictions in *DL-Lite* are usually given in this form, which is equivalent to $\exists \mathtt{HasOffState}.\top$.

6.1 Characterizing r-Satisfiablility

In this section, we propose a new characterization of r-satisfiability for a set $\mathcal{W}$ as in Lemma 3.13 which is tailored to the context of *DL-Lite*$^{\mathcal{H}}_{horn}$. As outlined in Section 3.3, the idea is to specify a polynomial amount of additional data that allows us to split the test proposed in Definition 3.12 into separate and independent consistency tests—of low complexity—focusing on single elements of $\mathcal{W}$, which may take this data into account. Given in this form, the additional data can then be guessed in a nondeterministic procedure deciding r-satisfiability.

Note that both the goal and basic approach are similar in the characterizations we developed for $\mathcal{EL}$ in Sections 4.4.2 and 5.1. But there are two main differences, especially to the latter one, which also targets TCQ entailment. Regarding *DL-Lite*$^{\mathcal{H}}_{horn}$, we also consider rigid roles, which requires us to considerably extend the additional data and conditions to be tested. Second, we apply the characterization also for investigating data complexity and hence cannot simplify the problem by neglecting the ABoxes in the TKB.[2]

In addition to the set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ that consists of worlds $W_1, \dots, W_k$, we regard a mapping $\iota\colon [0,n] \to [1,k]$. Observe that, for now, we assume both to be given. We however never consider $\mathcal{W}$ as a whole, but regard its elements $W_1, \dots, W_k$ independently of each other.

In what follows, we first introduce notions to describe the additional data and to specify the conditions, then present the characterization itself, and lastly prove its correctness.

6.1.1 ABox Types, Consequences, and Witness Queries

As described in Section 3.3, the goal of including the additional data is to simulate the effects of the shared domain of the exponentially many interpretations in Definition 3.12 (Functions (F1) and (F2)) so that we can focus on single interpretations.

Regarding Function (F1), our tests use an *ABox type* for *DL-Lite*, a set of (negated) assertions that is to be satisfied by all of the interpretations.

Definition 6.1 (ABox Type) An *ABox type* for $\mathcal{K}$ is a set

$$\begin{aligned}\mathcal{A}_{\mathsf{R}} \subseteq \{&B(a), \neg B(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}),\ B \in \mathbb{B}_{\mathsf{R}}(\mathcal{O})\} \cup \\ &\{R(a,b), \neg R(a,b) \mid a,b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}),\ R \in \mathsf{N}^{-}_{\mathsf{RR}}(\mathcal{O})\}\end{aligned}$$

with the property that $\alpha \in \mathcal{A}_{\mathsf{R}}$ iff $\neg\alpha \notin \mathcal{A}_{\mathsf{R}}$. ♢

Note that this definition extends Definition 5.2, specifying the notion for TCQs and $\mathcal{EL}$, by additionally considering the individuals in the ABoxes and rigid roles.

[2] According to Lemma 3.7, the data given in the ABoxes can be encoded into the TCQ, and the ABoxes can then be considered to be empty. This approach does however not suit data complexity investigations since the TCQ then cannot be assumed to be independent of the data any more. That is, if something has to be decided w.r.t. the query (e.g., for the contained CQs) in order to answer it, then data complexity cannot be considered, but the usually higher combined complexity has to be applied instead.

Regarding Function (F2), first observe that the satisfiability of CQs occurring only in positive literals in the conjunctions χ_i with $i \in [1, k]$ cannot be contradicted by some interpretation—we focus on all $\mathcal{J}_j$ and $\mathcal{I}_j$ with $j \neq i$—if both the model $\mathcal{J}_i$ of the CQ and the other interpretation satisfy a common ABox type and the ontology $\mathcal{O}$. Since this is ensured in our tests, it remains to consider the negative CQ literals. It is easy to see that, if rigid roles are left apart, the satisfiability of such literals can only be contradicted (based on rigid symbols) by an interpretation satisfying $\mathcal{O}$ and a common ABox type through single domain elements, because we assume the CQs to be connected; note that this is the same w.r.t. $\mathcal{EL}$, where we disregard rigid roles entirely. Regarding rigid roles, we additionally have to consider structures formed by related individuals in a specific interpretation if they are invariant to time. In particular, we have to consider those that exist because we require the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ to satisfy certain ABox assertions and CQs contained in Φ. Since ABox types capture the rigid structures formed by the former, it remains to regard the CQs satisfied by some of the interpretations. Targeting such CQs, we define the *consequences* of a CQ, a set of (also negated) assertions which we then require to be satisfied by all of the interpretations.

In the following, for a CQ or set of CQs Q, $\mathcal{A}_Q$ denotes the ABox obtained from Q by instantiating all variables x in the CQs with fresh individual names of the form a_x.

Definition 6.2 (Consequences) The set $\mathcal{C}_\mathcal{O}(\varphi)$ of *consequences* of a CQ φ is defined as

$$\mathcal{C}_\mathcal{O}(\varphi) := \{C(a) \mid C \in \mathbb{B}^-_{\mathsf{R}}(\mathcal{O}),\ a \in \mathsf{N}_\mathsf{I}(\mathcal{A}_\varphi),\ \mathcal{O} \models \textstyle\bigsqcap \mathbb{B}(\mathcal{A}_\varphi, a) \sqsubseteq C\} \cup$$
$$\{R(a,b) \mid R \in \mathsf{N}^-_{\mathsf{RR}}(\mathcal{O}),\ S(a,b) \in \mathcal{A}_\varphi,\ \mathcal{O} \models S \sqsubseteq R\};$$

where $\mathbb{B}(\mathcal{A}, a) := \{A \in \mathsf{N}_\mathsf{C} \mid A(a) \in \mathcal{A}\} \cup \{\exists R \mid R \in \mathsf{N}^-_{\mathsf{R}},\ R(a,b) \in \mathcal{A}\}$. ♢

Note that we cannot exactly capture (this part of) the shared domain by simply requiring such assertions to be satisfied by all interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$; in particular, we cannot enforce the latter to instantiate the structures with the same kind of individuals (i.e., w.r.t. the new individual names a_x), where "kind" refers to the concepts and roles the individual instantiates. Nevertheless, in the context of *DL-Lite*$^\mathcal{H}_{horn}$, the ontology cannot express meaningful information regarding role successors either, since the logic does not allow to express qualified existential restriction. This means that the kind of related individuals cannot be a reason for contradictions caused by the ontology. And we later show that, if information on some $\mathcal{J}_j$ or $\mathcal{I}_j$ is relevant to test the satisfiability of a conjunction χ_i with $i \in [1, k]$, then it suffices to know about the *existence* of such rigid structures. This specifically means that TCQs and TKBs can enforce relations between structures existing at different time points only via named individuals (e.g., by requiring certain assertions or CQs to be satisfied).

The notion of consequences allows us to describe the time-invariant effects of satisfied CQs, but captures only extracts of the domain. The named individuals are covered already via ABox types and consequences.[3] We therefore additionally describe critical consequences of arbitrary, unnamed domain parts. Specifically, we characterize the

[3] Note that we actually still disregard one aspect of the relevant time-invariant information on named individuals, the case that a flexible relation implies several rigid ones to the same unnamed successor. This is covered in the next section.

satisfaction of CQs—those in the negative literals—independent of time in terms of other CQs, so-called *witness queries*. These are CQs without variables and, more importantly, of *tree-shape*. Observe that CQs of this shape sufficiently capture the unnamed part for the purpose of r-satisfiability, where we look for interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ that do (not) satisfy certain axioms and CQs. The resulting relations in the unnamed domain parts can be seen as trees since we consider a *DL-Lite*$^{\mathcal{H}}_{horn}$ ontology. If we disregard the named individuals, it is thus safe to focus on CQs of this shape.

We start defining what we consider as *tree-shaped* CQs. Note that the graphs of the CQs are generally not directed (trees), namely because *DL-Lite*$^{\mathcal{H}}_{horn}$ allows for inverse roles. We therefore consider functions similar to the *tree witnesses* defined in [Kon+10]. These functions describe bijections between the variables of a CQ φ and the nodes of a tree (i.e., the tree is similar to G_φ but directed) so that an atom $R(x, y) \in \varphi$ implies that the tree contains an edge between the corresponding nodes. We extend the functions to incorporate not only all atoms $R(x, y) \in \varphi$ but explicitly consider roles S for which the ontology entails an RI $S \sqsubseteq R$.

Definition 6.3 (Tree-Shaped) Let φ be a CQ with $\mathsf{N_I}(\varphi) = \emptyset$ and $x \in \mathsf{N_V}(\varphi)$.

A *tree witness* for φ (w.r.t. x and $\mathcal{O}$) is a function $\mathrm{f}_x \colon \mathsf{N_V}(\varphi) \to (\mathsf{N_R^-} \times 2^{\mathsf{N_R^-}})^*$ such that

- $\mathrm{f}_x(x) = \epsilon$;
- for all $\varrho \cdot (S, \mathcal{R}) \in \mathsf{range}(\mathrm{f})$ and $R \in \mathcal{R}$, we have $\mathcal{O} \models S \sqsubseteq R$; and
- for each $R(y, z) \in \varphi$, we have
  - $\mathrm{f}_x(z) = \mathrm{f}_x(y) \cdot (S_1, \mathcal{R})$ and some $S \in \mathcal{R}$ such that $\mathcal{O} \models S \sqsubseteq R$, or
  - $\mathrm{f}_x(y) = \mathrm{f}_x(z) \cdot (S_1, \mathcal{R})$ and some $S \in \mathcal{R}$ such that $\mathcal{O} \models S \sqsubseteq R^-$.

If a tree witness for φ exists, then we call φ *tree-shaped*.

For a tree-shaped CQ φ and a tree witness f_x for φ, the set $\mathrm{Con}_\mathcal{O}(\varphi, \mathrm{f}_x)$ is defined as the set of all sets $\mathcal{B} \subseteq \mathbb{B}(\mathcal{O})$ such that

- for each $A(y) \in \varphi$ with $\mathrm{f}_x(y) = \epsilon$, we have $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq A$;
- for each $(S, \mathcal{R}) \in \mathsf{range}(\mathrm{f}_x)$, we have $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S$;
- for each $A(y) \in \varphi$ with $\mathrm{f}_x(y) = \varrho \cdot (S, \mathcal{R})$, we have $\mathcal{O} \models \exists S^- \sqsubseteq A$; and
- for each $\varrho \cdot (S_1, \mathcal{R}_1) \cdot (S_2, \mathcal{R}_2) \in \mathsf{range}(\mathrm{f}_x)$, we have $\mathcal{O} \models \exists S_1^- \sqsubseteq \exists S_2$. ♢

Regarding tree witnesses, recall that we assume CQs to be connected. The last condition in the definition thus leads to the fact that the sets $\mathcal{R} \subseteq 2^{\mathsf{N_R^-}}$ are never empty.

Intuitively, the set $\mathrm{Con}_\mathcal{O}(\varphi, \mathrm{f}_x)$ contains all sets $\mathcal{B}$ of basic concepts that, if satisfied in an interpretation $\mathcal{I}$ by one of its domain elements, imply the satisfaction of φ. Note that the last two conditions for the sets $\mathcal{B}$ do not refer to these sets specifically; they ensure that the sets describe the whole query. Observe that the set $\mathrm{Con}_\mathcal{O}(\varphi, \mathrm{f}_x)$ is empty if they are not fulfilled w.r.t. the considered tree witness f_x. For simplicity, we may omit the variable x if we regard a tree witness f_x where x is irrelevant or clear from context and denote it by f, instead.

Tree witnesses can be used to describe CQs that may be satisfied by unnamed elements in models of $\mathcal{O}$, but their definition still focuses on a specific time point (i.e., it does not distinguish rigid and flexible symbols). Next, we consider the time-invariant setting. We specify *witness queries* for a tree-shaped CQ: CQs that contain only rigid symbols and, if satisfied in a model of $\mathcal{O}$, imply the satisfaction of the latter query at arbitrary time points. They are based on *witnesses*: sets of rigid basic concepts that, if satisfied in a model of $\mathcal{O}$, imply the satisfaction of a single basic concept at arbitrary time points; alternatively, they can characterize the existence of a specific element in canonical interpretations.

Definition 6.4 (Witness) A set $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ is a *witness* of a basic concept $B \in \mathbb{B}(\mathcal{O})$ w.r.t. $\mathcal{O}$ if there are $R_0, \dots, R_\ell \in \mathsf{N}_{\mathsf{R}}^-$ such that $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R_0$, $\mathcal{O} \models \exists R_{i-1}^- \sqsubseteq \exists R_i$ for all $i \in [1, \ell]$, and $\mathcal{O} \models \exists R_\ell^- \sqsubseteq B$.

Let further $\mathcal{I}$ be the canonical interpretation for a knowledge base $\langle \mathcal{O}, \mathcal{A} \rangle$, where $\mathcal{A}$ is an arbitrary ABox. Then, $\mathcal{B}$ is a *witness* of an element $u_{\varrho R_0 \dots R_\ell} \in \Delta_{\mathsf{u}}^{\mathcal{I}}$ w.r.t. $\langle \mathcal{O}, \mathcal{A} \rangle$ if $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R_0$ and $u_\varrho \in (\bigsqcap \mathcal{B})^{\mathcal{I}}$ or $\varrho \in \mathsf{N}_{\mathsf{I}}(\mathcal{A}) \cap (\bigsqcap \mathcal{B})^{\mathcal{I}}$.

The set of all witnesses of a basic concept or unnamed element X w.r.t. $\mathcal{O}$ is denoted by $\mathcal{W}_{\mathcal{O}}(X)$. For all $e \in \Delta^{\mathcal{I}} \setminus \Delta_{\mathsf{u}}^{\mathcal{I}}$, we define $\mathcal{W}_{\mathcal{O}}(e) = \emptyset$. ♢

Note that this definition corresponds to the one for $\mathcal{EL}$. Here, we however have to consider inverse roles and explicitly require the concepts C_i from Definition 5.4 to be of the form $\exists R_i^-$. Furthermore, it is not sufficient to focus on sets of concepts to describe the satisfaction of tree-shaped CQs through rigid symbols in the unnamed domain parts of interpretations since we consider rigid roles.

We therefore characterize the satisfaction of a tree-shaped CQ φ by *witness queries*, of which we have two kinds. If they are satisfied in a model of $\mathcal{O}$ through some homomorphism, *tree witness queries* imply the satisfaction of φ based on that homomorphism (i.e., at arbitrary time points). The other, simpler kind only implies the satisfaction of φ in general.

Definition 6.5 (Witness Query) Let f be a tree witness for a (tree-shaped) CQ φ w.r.t. $\mathcal{O}$, and let ψ be a CQ containing only variables of the form x_ϱ, where $\varrho \in \mathsf{range}(\mathrm{f})$.

An element $\varrho \in \mathsf{range}(\mathrm{f})$ is *rigidly witnessed* in ψ (w.r.t. f) if $\varrho \neq \epsilon$ and one of the following holds:

- $\varrho = \sigma \cdot (S, \mathcal{R})$ and there is a set $\mathcal{B}_{\exists S} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B}_{\exists S} \sqsubseteq \exists S$ and $\mathcal{B}_{\exists S}(x_\sigma) \subseteq \psi$.[4]
- $\varrho = \sigma \cdot (S, \mathcal{R})$ and σ is rigidly witnessed in ψ.

Given a set $\mathcal{B} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ with subset $\mathcal{B}|_{\mathsf{R}} := \mathcal{B} \cap \mathbb{B}_{\mathsf{R}}(\mathcal{O})$, ψ is a *tree witness query* for φ (w.r.t. $\mathcal{O}$, $\mathcal{B}$, and f) if it is minimal (w.r.t. set inclusion regarding the set of atoms) among all CQs satisfying the following conditions:

- $\mathcal{B}|_{\mathsf{R}}(x_\epsilon) \subseteq \psi$;
- for each $A(y) \in \varphi$ with $\mathrm{f}(y) = \epsilon$, we have (i) $\mathcal{O} \models \bigsqcap \mathcal{B}|_{\mathsf{R}} \sqsubseteq A$, or (ii) $A \in \mathsf{N}_{\mathsf{RC}}$ and $A(x_\epsilon) \in \psi$;

[4] Note that $\sigma = \epsilon$ is possible.

- for each $A(y) \in \varphi$ with $\mathrm{f}(y) = \varrho \cdot (S, \mathcal{R})$, we have (i) $\mathrm{f}(y)$ is rigidly witnessed in ψ, or (ii) there is a set $\mathcal{B}_A \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ with $\mathcal{O} \models \bigsqcap \mathcal{B}_A \sqsubseteq A$ and $\mathcal{B}_A(x_{\mathrm{f}(y)}) \subseteq \psi$;
- for each $\varrho \cdot (S, \mathcal{R}) \in \mathsf{range}(\mathrm{f})$, we have (i) $\varrho \cdot (S, \mathcal{R})$ is rigidly witnessed in ψ, or (ii) $\mathcal{R} \subseteq \mathsf{N}_{\mathsf{RR}}$ and $R(x_\varrho, x_{\varrho \cdot (S,\mathcal{R})}) \in \psi$ for all $R \in \mathcal{R}$.

Further, ψ is a *witness query* for φ (w.r.t. $\mathcal{O}$) if

- either ψ is a tree witness query for φ w.r.t. $\mathcal{O}$, or
- $\psi = \exists x.\mathcal{B}(x)$ for some $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that there are an $R \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O})$ and tree witness f for φ w.r.t. $\mathcal{O}$ such that $\mathcal{B}$ is a witness of $\exists R$ w.r.t. $\mathcal{O}$ and $\{\exists R\} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$. $\diamondsuit$

For technical reasons, we in the following assume that x_ϵ occurs in every tree witness query. This is without loss of generality and can be achieved, for instance, by considering a fresh concept name that subsumes $\top$. We next prove that the satisfaction of a witness query for a CQ φ implies the satisfaction of φ.

Lemma 6.6 *Given an interpretation $\mathcal{I}$ that is a model of $\mathcal{O}$ and a witness query ψ for a CQ φ, we have that $\mathcal{I} \models \psi$ implies $\mathcal{I} \models \varphi$.*

Proof. We assume π to be the homomorphism of ψ into $\mathcal{I}$ (*) and, in line with Definition 6.5, consider two cases.

(I) Let ψ be a tree witness query w.r.t. a set $\mathcal{B}$ and tree witness f; note that we have $\mathcal{B} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$. The goal is to define a homomorphism π' of φ into $\mathcal{I}$. We define π' based on an auxiliary mapping $\tau \colon \mathsf{range}(\mathrm{f}) \to \Delta^{\mathcal{I}}$ as follows: $\pi'(y) := \tau(\mathrm{f}(y))$, for all $y \in \mathsf{N}_{\mathsf{V}}(\varphi)$. Next, we 1) specify τ and then 2) show that π' is indeed a homomorphism of φ into $\mathcal{I}$.

1) For all $\varrho \in \mathsf{range}(\mathrm{f})$ that are not rigidly witnessed in ψ, we define $\tau(\varrho) := \pi(x_\varrho)$. This especially applies to ϵ. Note that x_ϵ and all x_ϱ must occur in ψ, by our assumption on x_ϵ and Definition 6.5; for the latter, this follows from (ii) in the last condition in the definition of tree witness query where we have $\mathcal{R} \neq \emptyset$ by Definition 6.3.

For the remaining elements in $\mathsf{range}(\mathrm{f})$, the definition can thus be given by induction over the structure of f. Since they are rigidly witnessed, which is never the case for ϵ, we can consider them to be of the form $\varrho \cdot (S, \mathcal{R}) \in \mathsf{range}(\mathrm{f})$; $\varrho = \epsilon$ is possible. In particular, our definition will be such that $(\tau(\varrho), \tau(\varrho \cdot (S, \mathcal{R}))) \in S^{\mathcal{I}}$. In the base case, we assume $\varrho \cdot (S, \mathcal{R})$ to be *directly* rigidly witnessed, which means that no prefix of ϱ is rigidly witnessed. Then, $\tau(\varrho)$ is defined—as $\pi(x_\varrho)$—, and there is a set $\mathcal{B}_{\exists S} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B}_{\exists S} \sqsubseteq \exists S$ and $\mathcal{B}_{\exists S}(x_\varrho) \subseteq \psi$. The latter and (*) yield $\pi(x_\varrho) \in (\bigsqcap \mathcal{B}_{\exists S})^{\mathcal{I}}$, and $\mathcal{I} \models \mathcal{O}$ then implies $\pi(x_\varrho) \in (\exists S)^{\mathcal{I}}$. Hence, there must exist an element $e \in \Delta^{\mathcal{I}}$ such that $(\tau(\varrho), e) \in S^{\mathcal{I}}$, and we can set $\tau(\varrho \cdot (S, \mathcal{R})) := e$.

For the induction step, we consider elements of the form $\varrho \cdot (S_1, \mathcal{R}_1) \cdot (S_2, \mathcal{R}_2)$, assume $\varrho \cdot (S_1, \mathcal{R}_1)$ to be rigidly witnessed, and $\tau(\varrho \cdot (S_1, \mathcal{R}_1))$ to be defined; note that $x_{\varrho \cdot (S_1,\mathcal{R}_1) \cdot (S_2,\mathcal{R}_2)}$ does not necessarily occur in ψ. The (last condition in the) definition of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ (see Definition 6.3), then yields that $\mathcal{O} \models \exists S_1^- \sqsubseteq \exists S_2$. We have $\tau(\varrho \cdot (S_1, \mathcal{R}_1)) \in (\exists S_1^-)^{\mathcal{I}}$ by our definition of τ and hence get $\tau(\varrho \cdot (S_1, \mathcal{R}_1)) \in (\exists S_2)^{\mathcal{I}}$,

because $\mathcal{I} \models \mathcal{O}$. So there is an S_2-successor e of $\tau(\varrho \cdot (S_1, \mathcal{R}_1))$ in $\mathcal{I}$, and we can set $\tau(\varrho \cdot (S_1, \mathcal{R}_1) \cdot (S_2, \mathcal{R}_2)) := e$.

2) We consider the definition of a tree witness query and first regard an atom $A(y) \in \varphi$.

- If $\mathrm{f}(y) = \epsilon$, then either (i) $A \in \mathsf{N_{RC}}$ and $A(x_\epsilon) \in \psi$, or (ii) $\mathcal{O} \models \bigsqcap \mathcal{B}|_{\mathsf{R}} \sqsubseteq A$. From (*) and, for (ii), $\mathcal{I} \models \mathcal{O}$, we get $\pi(x_\epsilon) \in A^{\mathcal{I}}$, in both cases. Given that our definition is such that $\pi'(y) = \tau(\epsilon) = \pi(x_\epsilon)$, this means $\pi'(y) \in A^{\mathcal{I}}$.

- If $\mathrm{f}(y)$ is of the form $\varrho \cdot (S_1, \mathcal{R}_1) \cdot (S_2, \mathcal{R}_2)$, then either (i) $\mathrm{f}(y)$ is rigidly witnessed in ψ, or (ii) there is a set $\bigsqcap \mathcal{B}_A \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that $\mathcal{O} \models \mathcal{B}_A \sqsubseteq A$ and $\mathcal{B}_A(x_{\mathrm{f}(y)}) \subseteq \psi$. Regarding (i), we have $\tau(\mathrm{f}(y)) \in (\exists S_2^-)^{\mathcal{I}}$ and hence $\pi'(y) \in (\exists S_2^-)^{\mathcal{I}}$, both by our construction. From the third condition in the definition of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ and $\mathcal{I} \models \mathcal{O}$, we then get $\pi'(y) \in A^{\mathcal{I}}$.

We regard a role atom $R(y, z) \in \varphi$. According to Definition 6.3, we have one of the following:

- $\mathrm{f}(z) = \mathrm{f}(y) \cdot (S_1, \mathcal{R}_1)$ and there is an $S \in \mathcal{R}_2$ such that $\mathcal{O} \models S \sqsubseteq R$: Similar as above we then either have that (i) $\mathrm{f}(y) \cdot (S_1, \mathcal{R}_1)$ is rigidly witnessed in ψ, or (ii) $S(x_{\mathrm{f}(y)}, x_{\mathrm{f}(z)}) \in \psi$. For (i), observe that our definition of π' is (then) such that $(\pi'(y), \pi'(z)) \in S_1^{\mathcal{I}}$. Given $S_1 \sqsubseteq S$, by Definition 6.3, and $\mathcal{I} \models \mathcal{O}$, we thus get $(\pi'(y), \pi'(z)) \in R^{\mathcal{I}}$. Regarding (ii), (*) yields that $(\pi(x_{\mathrm{f}(y)}), \pi(x_{\mathrm{f}(z)})) \in S^{\mathcal{I}}$. The fact that π' is defined such that $(\pi'(y), \pi'(z)) = (\pi(x_{\mathrm{f}(y)}), \pi(x_{\mathrm{f}(z)}))$ and $\mathcal{I} \models \mathcal{O}$ then directly imply $(\pi'(y), \pi'(z)) \in R^{\mathcal{I}}$.

- $\mathrm{f}(y) = \mathrm{f}(z) \cdot (S_1, \mathcal{R}_1)$ and some $S \in \mathcal{R}_2$ such that $\mathcal{O} \models S \sqsubseteq R^-$: this case follows by dual arguments, exchanging $\mathrm{f}(y)$ and $\mathrm{f}(z)$, and replacing R by R^-.

This shows that π' is a homomorphism of φ into $\mathcal{I}$.

(II) Regarding the second case of Definition 6.5, we assume ψ to be of the form $\psi = \exists x.\mathcal{B}(x)$ for some $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that there are an $R \in \mathsf{N}_{\mathsf{R}}^-(\mathcal{O})$ and a tree witness f for ψ w.r.t. $\mathcal{O}$ such that $\mathcal{B}$ is a witness of $\exists R$ w.r.t. $\mathcal{O}$ and $\{\exists R\} \in \mathrm{Con}_{\mathcal{O}}(\psi, \mathrm{f})$. Given the assumptions that $\mathcal{I} \models \psi$ and $\mathcal{I} \models \mathcal{O}$, there must then be an element e that satisfies $\exists R$ in $\mathcal{I}$ by Definition 6.4. We proceed in two parts 1) and 2) as above.

1) We define π' based on a mapping τ analogously to the previous case, but begin by setting $\tau(\epsilon) := e$. The rest of τ is defined by induction similar to the above induction, by treating ϵ as if it was directly rigidly witnessed. Then, the only difference is that, in the base case, we cannot assume the element considered to be of the form $\varrho \cdot (S, \mathcal{R})$ and an instance of the concept $\exists S^-$ in $\mathcal{I}$, but have to regard ϵ. But since $\pi'(\epsilon)$ satisfies $\exists R$ by definition, $\{\exists R\} \in \mathrm{Con}_{\mathcal{O}}(\psi, \mathrm{f})$, and $\mathcal{I} \models \mathcal{O}$, we obtain the required S-successors by the second condition in the definition of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$.

2) Since all elements can be treated as rigidly witnessed this Part 2) is a special case of the above Part 2), except for the case of a concept atom $A(y) \in \varphi$ with $\mathrm{f}(y) = \epsilon$. But then we know that $\pi'(y) \in (\exists R)^{\mathcal{I}}$, and hence $\pi'(y) \in A^{\mathcal{I}}$ by the first condition in the definition of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$. □

The lemma shows that we have to especially focus on the satisfaction of witnesses of a CQ if we do not want the CQ to be satisfied at an arbitrary time point through unnamed domain parts.

Altogether, the definitions provided in this section enable us to characterize the Functions (F1) and (F2) of the shared domain from Definition 3.12 in an alternative way and hence to look for the interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ separately. In order to ensure a certain agreement between them on the interpretation of the rigid symbols, we next collect all the data important for the time-invariant setting in *r-complete tuples*.

6.1.2 r-Complete Tuples

In this section, we specify the additional data in the form of tuples and propose the property of *r-completeness* for characterizing r-satisfiablility. Recall the goal: independent and separate consistency tests for each $W \in \mathcal{W}$ and each $W_{\iota(i)}$ with $i \in [0, n]$, including the corresponding ABox $\mathcal{A}_i$, that may take the additional data into account and altogether decide if $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{O}$.

The additional data is a tuple of the form $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg}, R_{\mathsf{F}})$, where

- $\mathcal{A}_{\mathsf{R}}$ is an ABox type for $\mathcal{K}$,
- $Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg} \subseteq \mathcal{Q}_{\Phi}$, and
- $R_{\mathsf{F}} \subseteq \{\exists S(b) \mid S \in \mathsf{N}_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}_{\mathsf{RR}},\ b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}\}$

where the set $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}} \subseteq \mathsf{N}_{\mathsf{I}}$ contains an individual name a_x for each variable x occurring in a CQ in Q_{R}. Note that, because of our assumption that the CQs in Φ have no variables in common, each $a_x \in \mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ can be unambiguously associated to a CQ containing x.

As outlined above, the idea is to fix the interpretation of the named individuals via $\mathcal{A}_{\mathsf{R}}$, to consider the consequences of the CQs satisfied at some time point and collected in Q_{R}, and to regard the witness queries of the CQs not satisfied at *some* time point and collected in $Q_{\mathsf{R}}^{\neg}$. Observe that Q_{R} and $Q_{\mathsf{R}}^{\neg}$ may overlap. The set R_{F} is necessary to capture possible effects of RIs of $\mathcal{O}$, as it is sketched in the following example. The additional data thus consists of a number of assertions and queries that is polynomial in the size of Φ.

Example 6.7 At $n = 1$, the TCQ

$$\Phi := (\bigcirc_P A(a)) \wedge \neg\,(\exists x.R_1(a, x) \wedge R_2(a, x))$$

is not satisfiable w.r.t. the TKB $\mathcal{K}\ \langle \mathcal{O}, (\emptyset, \emptyset)\rangle$, where $\mathcal{O}$ contains the CIs $A \sqsubseteq \exists S$, $S \sqsubseteq R_1$, and $S \sqsubseteq R_2$, and R_1 and R_2 are the only rigid symbols.

This is because every model $\mathfrak{I} = \mathcal{I}_0, \mathcal{I}_1, \dots$ of $\mathcal{K}$ and Φ has to be such that all of $\mathcal{I}_0, \mathcal{I}_1, \dots$ satisfy $\mathcal{O}$, $\mathcal{I}_0 \models A(a)$, and $\mathcal{I}_1 \not\models \exists x.R_1(a, x) \wedge R_2(a, x)$ (see Definitions 3.3 and 3.5). Then, there must be an element e such that the tuple (a, e) is contained in $S^{\mathcal{I}_0}, R_1^{\mathcal{I}_0}$, and $R_2^{\mathcal{I}_0}$. The fact that $\mathfrak{I}$ respects the rigid names then yields that, for all $i \geq 0$, the tuple has to be contained in $R_1^{\mathcal{I}_i}$ and $R_2^{\mathcal{I}_i}$. Hence, $\mathcal{I}_1 \models \exists x.R_1(a, x) \wedge R_2(a, x)$.

ABox types and consequences however cannot capture such a scenario where $\mathcal{K}$ and Φ imply that a named individual has a flexible role successor and that relation implies several rigid relations, since the latter then target the same unnamed successor element.

That is, while the TCQ has no model, there may be interpretations as in Definition 3.12 except that they do not share one domain, but do satisfy a common ABox type and consequences of φ_1 and φ_2.

Consider $W_1 = \{p_1\}$, $W_2 = \emptyset$, a mapping $\iota = \{0 \mapsto 1, 1 \mapsto 2\}$, and arbitrary interpretations $\mathcal{J}_1, \mathcal{I}_2, \mathcal{I}_0, \mathcal{I}_1$ as in Definition 3.12 except that they do not share one domain. We assume $\varphi_1 = A(a)$. Definition 3.12 requires φ_1 (because $W_1 = \{p_1\}$) to be satisfied by $\mathcal{J}_1$ and $\mathcal{I}_0$.

Obviously, there is an ABox type $\mathcal{A}_{\mathsf{R}} = \{\exists R_1(a), \exists R_2(a)\}$ for $\mathcal{K}$, and we have the consequences $\mathcal{C}_{\mathcal{O}}(\varphi_1) = \{\exists R_1(a), \exists R_2(a)\}$. Regarding W_2, we do not have to consider consequences. Observe that we can still find interpretations $\mathcal{J}_1, \mathcal{I}_2, \mathcal{I}_0, \mathcal{I}_1$ as above that additionally satisfy $\mathcal{A}_{\mathsf{R}}$ and the set of consequences. ♢

To simplify the presentation of the consistency tests, we define, for all $i \in [1, k]$, the sets $Q_i := \{\varphi_j \mid p_j \in W_i\}$ and $\mathcal{A}_{n+i} := \emptyset$ and extend ι such that $\iota(n + i) := i$. Further, we represent most of the data to be considered for a given tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg}, R_{\mathsf{F}})$ in the form of ABoxes as follows:

- $\mathcal{A}_{Q_{\mathsf{R}}} := \bigcup_{\varphi \in Q_{\mathsf{R}}} \mathcal{C}_{\mathcal{O}}(\varphi)$.

  We assume that the instantiation of the variables is done using the corresponding individual names from $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$.

- $\mathcal{A}_{Q_i}$ constitutes the ABox obtained from Q_i by instantiating all variables with the corresponding names from $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$.

- $\mathcal{A}_{R_{\mathsf{F}}} := \bigcup_{\exists S(b) \in R_{\mathsf{F}}} \mathcal{A}_{\exists S(b)}$, where $\mathcal{A}_{\exists S(b)}$ is constructed as follows:

  1. For every domain element $u_{b\varrho}$ of the canonical interpretation $\mathcal{I}_{\langle \mathcal{O}, \{\exists S(b)\}\rangle}$ with $|\varrho| \leq \max\{|\mathsf{N}_{\mathsf{V}}(\varphi)| \mid \varphi \in \mathcal{Q}_\Phi\}$, introduce a new individual name $a_{b\varrho}$.

     These new individual names are collected in the set $\mathsf{N}_{\mathsf{I}}^{\mathsf{tree}}$.

  2. For every $a_{\varrho S} \in \mathsf{N}_{\mathsf{I}}^{\mathsf{tree}}$, add the following assertions to the set $\mathcal{A}_{\exists S(b)}$, which is assumed to be empty initially:
     - for every $B \in \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ with $\mathcal{O} \models \exists S^- \sqsubseteq B$, the concept assertion $B(a_{\varrho S})$;
     - for every $R \in \mathsf{N}_{\mathsf{RR}}^{-}(\mathcal{O})$ with $\mathcal{O} \models S \sqsubseteq R$, the role assertion $R(a_\varrho, a_{\varrho S})$ if $\varrho \notin \mathsf{N}_{\mathsf{I}}(\mathcal{K})$, otherwise $R(\varrho, a_{\varrho S})$.

  Observe that these consequences only have to be considered up to a depth which ensures that possible and relevant matches (i.e., those that include the named individual b) of relevant CQs (those in $\mathcal{Q}_\Phi$) depending on the existence of some S-successor of b can be fully characterized. Note that the ABox $\mathcal{A}_{R_{\mathsf{F}}}$ defined in this way is of exponential size. But we show in Sections 6.2 and 6.4 that this does not affect our complexity results

Observe that, for $i, j \in [1, k]$, the ABoxes $\mathcal{A}_{Q_{\mathsf{R}}}$, $\mathcal{A}_{Q_i}$, and $\mathcal{A}_{R_{\mathsf{F}}}$ may share some individual names from $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$ and $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$, whereas different ABoxes $\mathcal{A}_{Q_i}$ and $\mathcal{A}_{Q_j}$ do not share any individual names from $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ since we assume the CQs to use disjoint variables.

The conditions we then test regarding a given tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg}, R_{\mathsf{F}})$ are captured by the property of *r-completeness* (see also the corresponding Definitions 4.13 and 5.6 for $\mathcal{EL}$).

Definition 6.8 (r-complete) A tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg}, R_{\mathsf{F}})$ as specified above is *r-complete* (w.r.t. $\mathcal{W}$ and ι) if the following hold for all $i \in [0, n+k]$:

(C1) $\mathcal{K}^i_{\mathsf{R}} := \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_{R_{\mathsf{F}}} \cup \mathcal{A}_i \rangle$ is consistent.

(C2) For all $p_j \in \overline{W_{\iota(i)}}$, we have $\mathcal{K}^i_{\mathsf{R}} \not\models \varphi_j$.

(C3) For all $W \in \mathcal{W}$ and $p_j \in W$, we have $\varphi_j \in Q_{\mathsf{R}}$.

(C4) For all $W \in \mathcal{W}$ and $p_j \in \overline{W}$, we have $\varphi_j \in Q_{\mathsf{R}}^{\neg}$.

(C5) For all CQs $\varphi \in Q_{\mathsf{R}}^{\neg}$ and witness queries ψ for φ w.r.t. $\mathcal{O}$, we have $\mathcal{K}^i_{\mathsf{R}} \not\models \psi$.[5]

(C6) For all $S \in \mathsf{N}_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}_{\mathsf{RR}}$ and $b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$, we have $\exists S(b) \in R_{\mathsf{F}}$ iff there is an index $j \in [0, n+k]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(j)}} \cup \mathcal{A}_j \rangle \models \exists S(b)$. ♢

The first two conditions together ensure that, for all considered worlds W_i ($W_{\iota(i)}$) with $i \in [1, k]$ ($i \in [0, n]$), exactly the queries specified by the world *can* be satisfied w.r.t. $\mathcal{O}$ if the assertions in $\mathcal{K}^{i+n}_{\mathsf{R}}$ ($\mathcal{K}^i_{\mathsf{R}}$) are taken into account. In line with the above described intention, the third (fourth) condition then makes sure that only the queries from Q_{R} ($Q_{\mathsf{R}}^{\neg}$) can occur in some $W \in \mathcal{W}$ (in some $\overline{W}$ where $W \in \mathcal{W}$). Condition (C5) ensures that the queries induced by some such set $\overline{W}$ are not entailed based on the rigid names, by requiring that the canonical models of all $\mathcal{K}^i_{\mathsf{R}}$ do not satisfy any of the witnesses of the tree-shaped queries in $Q_{\mathsf{R}}^{\neg}$ (see Lemma 2.13). And the last condition ensures that R_{F} is minimal; that is, the rigid consequences of $\exists S(b)$ are only considered if they are implied by an adapted KB $\mathcal{K}^i_{\mathsf{R}}$, excluding $\mathcal{A}_{R_{\mathsf{F}}}$.

We next show that the existence of an r-complete tuple w.r.t. $\mathcal{W}$ fully characterizes the r-satisfiability of $\mathcal{W}$.

Lemma 6.9 *$\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$ iff there is an r-complete tuple w.r.t. $\mathcal{W}$ and ι.*

The proof of the lemma is split over the following two subsections.

If $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$, then there is an r-complete tuple w.r.t. $\mathcal{W}$ and ι.

Let $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ be the interpretations over a domain Δ that exist according to the r-satisfiability of $\mathcal{W}$ (see Definition 3.12). We assume w.l.o.g. that Δ contains $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$ and that all individual names are interpreted as themselves in all these interpretations.

We first define the tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg}, R_{\mathsf{F}})$ as follows:

$$\begin{aligned}
\mathcal{A}_{\mathsf{R}} := {} & \{B(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}), B \in \mathbb{B}_{\mathsf{R}}(\mathcal{O}), a^{\mathcal{J}_1} \in B^{\mathcal{J}_1}\} \cup \\
& \{\neg B(a) \mid a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}), B \in \mathbb{B}_{\mathsf{R}}(\mathcal{O}), a^{\mathcal{J}_1} \notin B^{\mathcal{J}_1}\} \cup \\
& \{R(a,b) \mid a, b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}), R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O}), (a,b) \in R^{\mathcal{J}_1}\} \cup \\
& \{\neg R(a,b) \mid a, b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}), R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O}), (a,b) \notin R^{\mathcal{J}_1}\}; \\
Q_{\mathsf{R}} := {} & \{\alpha_j \in \mathcal{Q}_\Phi \mid W \in \mathcal{W}, p_j \in W\}; \\
Q_{\mathsf{R}}^{\neg} := {} & \{\alpha_j \in \mathcal{Q}_\Phi \mid W \in \mathcal{W}, p_j \notin W\};
\end{aligned}$$

[5] Observe that the condition is only relevant for tree-shaped CQs since there are no witness queries for CQs that are not of tree shape.

$$R_{\mathsf{F}} := \{\exists S(b) \mid S \in \mathsf{N}_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}_{\mathsf{RR}},\ b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \mathsf{N}_{\mathsf{I}}^{\mathsf{aux}},\ i \in [0, n+k],\\ \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \rangle \models \exists S(b)\}.$$

We prove that the tuple is r-complete by showing that it satisfies all the conditions in Definition 6.8. It is easy to see that $\mathcal{A}_{\mathsf{R}}$ is an ABox type for $\mathcal{O}$, that Q_{R} satisfies Condition (C3), and that $Q_{\mathsf{R}}^{\neg}$ complies with Condition (C4). Moreover, the rather straightforward definition of the tuple seems to make it easy to show that each of the knowledge bases $\mathcal{K}_{\mathsf{R}}^{i}$ has a model (Condition (C1)) which neither satisfies a CQ from $\mathcal{Q}_{\Phi} \setminus Q_{\iota(i)}$ (Condition (C2)) nor a witness of the queries in $Q_{\mathsf{R}}^{\neg}$ (Condition (C5)), based on the given interpretations; but special attention needs to be given to the UNA and Condition (C6). The crucial point is that the given interpretations may satisfy CQs in a conjunction $\chi_{\iota(i)}$ ($\chi_{\iota(i)}$) by a homomorphism that can however not be applied to satisfy an ABox $\mathcal{A}_{Q_i}$ ($\mathcal{A}_{Q_{\iota(i)}}$), because π maps different variables to the same domain element, which are represented as named individuals from $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ in the ABox; the same applies to the ABox $\mathcal{A}_{R_{\mathsf{F}}}$ w.r.t. the elements from $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ and $\mathsf{N}_{\mathsf{I}}^{\mathsf{tree}}$. Regarding Condition (C6), observe that the "if" direction does not directly yield that *each* of the given interpretations satisfies the assertions in $\mathcal{A}_{R_{\mathsf{F}}}$.

The idea is therefore to extend the given interpretations $\mathcal{J}_i$ ($\mathcal{I}_i$) in two steps. First, we construct models $\mathcal{I}'_{i+n}$ ($\mathcal{I}'_i$) of $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i+n)}} \cup \mathcal{A}_{i+n} \rangle$ ($\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \rangle$) that interpret the elements of $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ and $\mathsf{N}_{\mathsf{I}}^{\mathsf{tree}}$ by using duplicates of elements from Δ. In order to overcome the issue regarding Condition (C6), we ensure that these interpretations still share one domain and also interpret the rigid symbols in the same way. This allows us to then adapt these interpretations in a second step regarding the elements from $\mathsf{N}_{\mathsf{I}}^{\mathsf{tree}}$ in order to get models of the respective knowledge bases $\mathcal{K}_{\mathsf{R}}^{i+n}$ ($\mathcal{K}_{\mathsf{R}}^{i}$), which include $\mathcal{A}_{R_{\mathsf{F}}}$.

For this extension, we consider different canonical interpretations for all CQs $\varphi \in Q_{\mathsf{R}}$ satisfied in one of the given interpretations: $\mathcal{I}_{\mathcal{K}_{R_{\mathsf{F}}}}$, the one of $\mathcal{K}_{\varphi} := \langle \mathcal{O}, \mathcal{A}_{R_{\mathsf{F}}} \rangle$; $\mathcal{I}_{\mathcal{K}_{\varphi}}$, the one of $\mathcal{K}_{\varphi} := \langle \mathcal{O}, \mathcal{A}_{\varphi} \rangle$; and $\mathcal{I}_{\mathcal{K}_{\varphi}}^{\mathsf{R}}$, which collects the rigid consequences of $\mathcal{I}_{\mathcal{K}_{\varphi}}$ and is inductively defined according to Definition 2.11 with the adaptation that all symbols $X \in \mathsf{N}_{\mathsf{C}} \cup \mathsf{N}_{\mathsf{R}}$ are initially interpreted as follows:

$$X^0 := \begin{cases} X^{\mathcal{I}_{\mathcal{K}_{\varphi}}} & \text{if } X \in \mathsf{N}_{\mathsf{RC}} \cup \mathsf{N}_{\mathsf{RR}}, \\ \emptyset & \text{otherwise.} \end{cases}$$

Being defined in this way, $\mathcal{I}_{\mathcal{K}_{\varphi}}^{\mathsf{R}}$ behaves exactly as $\mathcal{I}_{\mathcal{K}_{\varphi}}$ w.r.t. the rigid names, but the interpretations of the flexible names only contain those tuples that are implied by the rigid information. Note that the domain of $\mathcal{I}_{\mathcal{K}_{\varphi}}^{\mathsf{R}}$ is part of the one of $\mathcal{I}_{\mathcal{K}_{\varphi}}$ since all elements u_{ϱ} that would be created by the iteration in Definition 2.11 for $\mathcal{I}_{\mathcal{K}_{\varphi}}^{\mathsf{R}}$ are also created by the one for $\mathcal{I}_{\mathcal{K}_{\varphi}}$ and are hence contained in the initial interpretation above. We consider the interpretations $\mathcal{I}_{\mathcal{K}_{\varphi}}^{\mathsf{R}}$ to make sure that all interpretations we construct also satisfy $\mathcal{A}_{Q_{\mathsf{R}}}$ in consideration of the UNA (together with $\mathcal{O}$), even if they do not satisfy an ABox $\mathcal{A}_{\varphi}$.

Observe that $\mathcal{K}_\varphi$ is consistent since $\varphi \in Q_{\mathsf{R}}$ implies that there is an interpretation $\mathcal{I}_i$ or $\mathcal{J}_i$ that satisfies φ and $\mathcal{O}$ and is thus a model of $\mathcal{K}_\varphi$.[6] This also leads to the fact that all the given interpretations satisfy the rigid consequences of $\mathcal{A}_\varphi$ w.r.t. $\mathcal{O}$ (i.e., particularly $\mathcal{C}_\mathcal{O}(\varphi)$, see Definition 6.2) because, by assumption, they share one domain and respect the rigid names. We can state properties of all $\varphi_j \in Q_{\mathsf{R}}$ that are crucial for our construction:

- $\mathcal{I}_{\mathcal{K}_{\varphi_j}}$ can be homomorphically embedded into each $\mathcal{J}_i$ ($\mathcal{I}_i$) for which we have $p_j \in W_{\iota(n+i)}$ ($p_j \in W_{\iota(i)}$) since there must be a homomorphism of φ_j into $\mathcal{J}_i$ ($\mathcal{I}_i$) and this entails the existence of domain elements that satisfy at least the symbols satisfied by the elements of $\mathcal{I}_{\mathcal{K}_{\varphi_j}}$.

- $\mathcal{I}^{\mathsf{R}}_{\mathcal{K}_{\varphi_j}}$ can be homomorphically embedded into each $\mathcal{J}_i$ ($\mathcal{I}_i$), because there must be some $\ell \in [1,k]$ such that $\mathcal{J}_\ell$ satisfies φ_j (and $\mathcal{O}$), and the rigid consequences of φ_j, represented by $\mathcal{I}^{\mathsf{R}}_{\mathcal{K}_{\varphi_j}}$, are satisfied in all the given interpretations.

These facts imply that we can, as the first step, extend all given interpretations $\mathcal{J}_i$ with $i \in [1,k]$ ($\mathcal{I}_i$, $i \in [0,n]$) to models $\mathcal{I}'_{n+i}$ ($\mathcal{I}'_i$) of $\mathcal{A}_{Q_{\mathsf{R}}}$ and $\mathcal{A}_{Q_{\iota(n+i)}}$ ($\mathcal{A}_{Q_{\iota(i)}}$) because they already contain elements that behave in the same way—at least, regarding the symbols that need to be satisfied to obtain such models:

- The common domain Δ is extended by the union of the domains of all $\mathcal{I}_{\mathcal{K}_\varphi}$ with $\varphi \in Q_{\mathsf{R}}$ (which are also the domains of $\mathcal{I}^{\mathsf{R}}_{\mathcal{K}_\varphi}$). Note that these domains may overlap in N_I and $\mathsf{N}^{\mathsf{aux}}_\mathsf{I}$.

- The individual names from $\mathsf{N}^{\mathsf{aux}}_\mathsf{I}$ are interpreted as themselves.

- For each $j \in [1,m]$ and $p_j \in W_{\iota(i)}$, all symbols are, on the domain of $\mathcal{I}_{\mathcal{K}_{\varphi_j}}$, interpreted exactly as in $\mathcal{I}_{\mathcal{K}_{\varphi_j}}$. Note that there are no role connections between the old and the new domains except between $\mathsf{N}_\mathsf{I}(\mathcal{K})$ and the elements of $\mathsf{N}^{\mathsf{aux}}_\mathsf{I}$ (*).

- If $p_j \in \overline{W_{\iota(i)}}$ ($p_j \in \overline{W_i}$), then all symbols are, on the domain of $\mathcal{I}_{\mathcal{K}_{\varphi_j}}$, interpreted exactly as in $\mathcal{I}^{\mathsf{R}}_{\mathcal{K}_{\varphi_j}}$.

Recall that Δ is not yet complete; it is still to be extended by the domain of $\mathcal{I}_{\mathcal{K}_{R_\mathsf{F}}}$. Though, this definition meets our requirement that $\mathcal{I}'_i \models \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \rangle$ and is such that $\mathcal{I}'_i \models \chi_{\iota(i)}$ for all $i \in [0, n+k]$, which can be seen given the following observations:

- $\mathcal{I}'_i$ satisfies $\mathcal{O}$ and $\mathcal{A}_{\mathsf{R}}$. This is because, given (*), the interpretation of symbols on the unnamed elements in the original domain Δ does not change (i.e., the interpretation of basic concepts on these elements neither changes); further, the new domain elements do not exhibit new behavior that was not already present in $\mathcal{I}_i$ ($\mathcal{J}_i$); and the latter also implies that the interpretation of basic concepts on the elements of N_I does not change.

[6] If two variables are mapped by the homomorphism to the same domain element, we obtain a model respecting the UNA by creating a copy of this element that satisfies exactly the same concept names and participates in the same role connections as the original element.

- If $i \leq n$, then $\mathcal{I}'_i$ satisfies $\mathcal{A}_i$ because $\mathcal{I}_i \models \mathcal{A}_i$ by the same reasons as in the previous item. Otherwise, $\mathcal{A}_i$ is trivially satisfied.
- $\mathcal{I}'_i$ is a model of $\mathcal{A}_{Q_\mathsf{R}}$ since that ABox consists exactly of the ABoxes $\mathcal{C}_\mathcal{O}(\varphi)$ with $\varphi \in Q_\mathsf{R}$ and those are satisfied by the new domain elements because $\mathcal{I}^\mathsf{R}_{\mathcal{K}_\varphi}$ is part of $\mathcal{I}'_i$ and $\mathcal{I}^\mathsf{R}_{\mathcal{K}_\varphi} \models \mathcal{C}_\mathcal{O}(\varphi)$.
- Similarly, $\mathcal{I}'_i$ satisfies $\mathcal{A}_{Q_{\iota(i)}}$ since that ABox consist exactly of the ABoxes $\mathcal{A}_{\varphi_j}$ with $p_j \in W_{\iota(i)}$, satisfied by $\mathcal{I}_{\mathcal{K}_{\varphi_j}}$, which is part of $\mathcal{I}'_i$.
- For each $p_j \in \overline{W_{\iota(i)}}$, we get $\mathcal{I}'_i \not\models \varphi_j$ since any homomorphism of φ_j into $\mathcal{I}'_i$ would allow us to also find one into $\mathcal{I}_i$ or, if $i > n$, $\mathcal{J}_{i-n}$ which contradicts the respective assumptions that $\mathcal{I}_i \models \chi_{\iota(i)}$ and $\mathcal{J}_{i-n} \models \chi_{\iota(i)}$. Hence, we know that $\mathcal{I}'_i \models \chi_{\iota(i)}$.

We come to the second part. Since the interpretations are models of the respective knowledge bases $\langle \mathcal{O}, \mathcal{A}_\mathsf{R} \cup \mathcal{A}_{Q_\mathsf{R}} \cup \mathcal{A}_{Q_{\iota(n+i)}} \cup \mathcal{A}_{n+i} \rangle$ with $i \in [0, n+k]$, we get that every assertion $\exists S(b) \in R_\mathsf{F}$ is satisfied in one of them by the above definition of R_F. That interpretation thus also satisfies the rigid consequences described in $\mathcal{A}_{R_\mathsf{F}}$ since it is a model of $\mathcal{O}$. But then this holds for all the interpretations because they interpret the rigid symbols on the named elements in the common domain (i.e., those in $\Delta \cap \mathsf{N}_\mathsf{I}(\mathcal{K}) \cap \mathsf{N}_\mathsf{I}^\mathsf{aux}$) in the same way and satisfy $\mathcal{A}_\mathsf{R}$ and $\mathcal{A}_{Q_\mathsf{R}}$, which contain all the relevant rigid information. We can thus extend the domain Δ by the domain of $\mathcal{I}_{\mathcal{K}_{R_\mathsf{F}}}$ and all interpretations $\mathcal{I}'_{n+i}$ with $i \in [1, k]$ ($\mathcal{I}_i$ with $i \in [0, n]$) as follows. The names from $\mathsf{N}_\mathsf{I}^\mathsf{tree}$ are interpreted by themselves and, regarding the other names, these elements $a_{b\varrho} \in \mathsf{N}_\mathsf{I}^\mathsf{tree}$ are interpreted as the elements that already exist since they describe the consequences of assertions in R_F; note that we may again have to copy elements if the UNA would be violated otherwise. Note that this extension does not introduce new role connections between the old and the new domains except between $\mathsf{N}_\mathsf{I}(\mathcal{K})$ and the elements of $\mathsf{N}_\mathsf{I}^\mathsf{aux}$ and $\mathsf{N}_\mathsf{I}^\mathsf{tree}$, similar to (*). We therefore can argue similarly as above that the final interpretations $\mathcal{I}'_i$ are models as required for Condition (C1).

In what follows, we use these constructed interpretations to show that the tuple $(\mathcal{A}_\mathsf{R}, Q_\mathsf{R}, Q^\neg_\mathsf{R}, R_\mathsf{F})$, in addition to the above covered Conditions (C1), (C3), (C4), and (C6), also satisfies Conditions (C2) and (C5). This means it is r-complete.

Regarding Condition (C2), we assume that there are an $i \in [0, n+k]$ and a $p_j \in \overline{W_{\iota(i)}}$ such that $\mathcal{K}^i_\mathsf{R} \models \varphi_j$, which yields $\mathcal{I}'_i \models \varphi_j$. This directly contradicts the above shown fact that $\mathcal{I}'_i \models \chi_{\iota(i)}$.

The proof for Condition (C5) is also by contradiction. We assume that there are an index $i \in [0, n+k]$, a tree-shaped CQ $\varphi_j \in Q^\neg_\mathsf{R}$, and a witness query ψ for φ_j w.r.t. $\mathcal{K}^i_\mathsf{R}$ such that $\mathcal{K}^i_\mathsf{R} \models \psi$, and thus also $\mathcal{I}'_i \models \psi$ holds. However, $\varphi_j \in Q^\neg_\mathsf{R}$ yields that there is a $W_\ell \in \mathcal{W}$ with $\ell \in [1, k]$ such that $p_j \notin W_\ell$, and thus $\mathcal{I}'_{n+\ell} \not\models \varphi_j$. By Lemma 6.6, we know that $\mathcal{I}'_{n+\ell} \not\models \psi$. But this contradicts the facts that ψ contains only rigid names and that $\mathcal{I}'_{n+\ell}$ and $\mathcal{I}'_i$ respect the rigid names. This concludes the proof of the first direction of Lemma 6.9.

If there is an r-complete tuple w.r.t. $\mathcal{W}$ and ι, then $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$.

The proof of the converse direction is more involved; the general approach is again the same as for $\mathcal{EL}$. We assume an r-complete tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q^{\neg}_{\mathsf{R}}, R_{\mathsf{F}})$ to be given and, as before, follow the lines of Definition 3.12. The goal is thus to show that we can construct interpretations $\mathcal{J}_1, \dots, \mathcal{J}_k, \mathcal{I}_0, \dots, \mathcal{I}_n$ that share one domain and satisfy the other requirements of the definition. The idea is to integrate the canonical interpretations of the KBs in Condition (C1) of Definition 6.8 to construct these interpretations.

In what follows, we first provide auxiliary definitions, then define the interpretations, and lastly prove that they are as required. Note that, in the remainder of the proof, we generally do not reference Definition 6.8 explicitly if we refer to Conditions (C1)–(C6).

For all $i \in [0, n+k]$, consider the following definitions.

- $\mathcal{I}_i := \mathcal{I}_{\mathcal{K}^i_{\mathsf{R}}}$ denotes the canonical interpretation of the KB $\mathcal{K}^i_{\mathsf{R}}$.
- We define $\Delta^{\mathcal{I}_i}_{\mathsf{a}} := \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}} \cap \Delta^{\mathcal{I}_i}$ to distinguish the elements in the domain of $\mathcal{I}_i$ that are contained in $\mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$, and similarly write $\Delta^{\mathcal{I}_i}_{\mathsf{t}}$ and $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$ for the sets containing the elements from $\mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$ and, respectively, the unnamed domain elements that are unique to the canonical interpretation $\mathcal{I}_i$.
- We also write a^i_x for every element $a_x \in \Delta^{\mathcal{I}_i}_{\mathsf{a}}$, $a^i_{b\varrho}$ for every $a_{b\varrho} \in \Delta^{\mathcal{I}_i}_{\mathsf{t}}$, and u^i_ϱ for every $u_\varrho \in \Delta^{\mathcal{I}_i}_{\mathsf{u}}$. Further, for any element $e \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}} \cup \mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$, we may use e^i to denote the corresponding element in $\mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \Delta^{\mathcal{I}_i}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_i}_{\mathsf{t}}$; note that we thus consider $a^i := a$ for all $a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$.

Hence, the domain of $\mathcal{I}_i$ is composed of the pairwise disjoint sets $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$, $\Delta^{\mathcal{I}_i}_{\mathsf{a}}$, $\Delta^{\mathcal{I}_i}_{\mathsf{t}}$, and $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$. We state this fact for future reference.

Fact 6.10 *The set $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$, all $\Delta^{\mathcal{I}_i}_{\mathsf{a}}$ with $i \in [0, n+k]$, all $\Delta^{\mathcal{I}_i}_{\mathsf{t}}$ with $i \in [0, n+k]$, and all $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$ with $i \in [0, n+k]$ are pairwise disjoint.*

Next, we construct the interpretations $\mathcal{J}_0, \dots, \mathcal{J}_{n+k}$ as required for the r-satisfiability of $\mathcal{W}$; where $\mathcal{J}_0, \dots, \mathcal{J}_n$ represent $\mathcal{I}_0, \dots, \mathcal{I}_n$ of Definition 3.12 and $\mathcal{J}_{n+1}, \dots, \mathcal{J}_{n+k}$ represent $\mathcal{J}_1, \dots, \mathcal{J}_k$. To this end, we join the canonical interpretations $\mathcal{I}_i$. The idea is that, for all $i \in [0, n+k]$, $\mathcal{I}_i$ represents the parts specific to $\mathcal{J}_i$ and, for the interpretation of the rigid symbols in $\mathcal{J}_i$, all $\mathcal{I}_j$ with $j \in [0, n+k]$ are considered. The interpretation of the flexible symbols then can obviously not be solely based on $\mathcal{I}_i$ but has to be adjusted. In this way, we ensure that all of $\mathcal{J}_0, \dots, \mathcal{J}_{n+k}$ interpret the rigid concept names in the same way.

The common domain Δ is defined as follows:

$$\Delta := \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \bigcup_{i=0}^{n+k} \Delta^{\mathcal{I}_i}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_i}_{\mathsf{t}} \cup \Delta^{\mathcal{I}_i}_{\mathsf{u}}.$$

$\mathcal{J}_i$ is specified below, for all $i \in [0, n+k]$:

- For all $a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$: $a^{\mathcal{J}_i} := a$.
- For all rigid concept names A: $A^{\mathcal{J}_i} := \bigcup_{j=0}^{n+k} A^{\mathcal{I}_j}$.

- For all flexible concept names A:

$$A^{\mathcal{J}_i} := A^{\mathcal{I}_i} \cup \bigcup_{j=0}^{n+k} \bigcup_{\substack{\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O}), \\ \mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq A}} (\bigsqcap \mathcal{B})^{\mathcal{I}_j} \cup$$
$$\bigcup_{j=0}^{n+k} \{u^j_{\varrho R} \in \Delta_{\mathsf{u}}^{\mathcal{I}_j} \mid \mathcal{O} \models \exists R^- \sqsubseteq A, \mathcal{W}_{\mathcal{O}}(u^j_{\varrho R}) \neq \emptyset\}.$$

- For all rigid role names R: $R^{\mathcal{J}_i} := \bigcup_{j=0}^{n+k} R^{\mathcal{I}_j}$.
- For all flexible role names R:

$$R^{\mathcal{J}_i} := R^{\mathcal{I}_i} \cup \bigcup_{j=0}^{n+k} \bigcup_{\substack{S \in \mathsf{N}^-_{\mathsf{RR}}(\mathcal{O}) \\ \mathcal{O} \models S \sqsubseteq R}} S^{\mathcal{I}_j} \cup \bigcup_{j=0}^{n+k} \{(d,e) \in R^{\mathcal{I}_j} \mid \exists X \in \{d,e\}, \mathcal{W}_{\mathcal{O}}(X) \neq \emptyset\}.$$

We thus have constructed interpretations $\mathcal{J}_1, \ldots, \mathcal{J}_k$ that share the same domain and respect the rigid names since, for all $X \in \mathsf{N}_{\mathsf{RC}} \cup \mathsf{N}_{\mathsf{RR}}$ and $i \in [0, n+k]$, the definition of $X^{\mathcal{J}_i}$ is independent of i. Most parts of the specification are straightforward.

We next show connections between $\mathcal{J}_0, \ldots, \mathcal{J}_{n+k}$ and the canonical interpretations, which also help to clarify the picture of the former. In the remainder of this section, we then prove that the interpretations fulfill the other requirements for the r-satisfiability of $\mathcal{W}$.

Lemma 6.11 *For all $i \in [0, n+k]$, elements $d, e \in \Delta^{\mathcal{I}_i}$, and role names $R \in \mathsf{N}_{\mathsf{R}}$:*

$$(d,e) \in R^{\mathcal{J}_i} \text{ iff } (d,e) \in R^{\mathcal{I}_i}.$$

Proof. ($\Leftarrow$) This direction follows directly from the definition of $R^{\mathcal{J}_i}$.

($\Rightarrow$) We focus on the definition of $R^{\mathcal{J}_i}$. If R is flexible, we need to consider two cases; in both, we assume $i \neq j$: (i) $(d,e) \in S^{\mathcal{I}_j}$ for some rigid subrole S of R, and (ii) $(d,e) \in R^{\mathcal{I}_j}$ and either d or e has a witness w.r.t. $\mathcal{O}$. Given $d, e \in \Delta^{\mathcal{I}_i}$, Fact 6.10 however implies $d, e \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$ in both cases. Hence, (ii) is impossible since named domain elements cannot have witnesses according to Definition 6.4. Regarding (i), we get $S(d,e) \in \mathcal{A}_{\mathsf{R}}$ since $S \in \mathsf{N}_{\mathsf{RR}}$, that $\mathcal{A}_{\mathsf{R}}$ is an ABox type, and $\mathcal{I}_j \models \mathcal{A}_{\mathsf{R}}$. Then, $\mathcal{I}_i \models \mathcal{A}_{\mathsf{R}}$ implies $(d,e) \in S^{\mathcal{I}_i}$, and $\mathcal{I}_i \models \mathcal{O}$ yields $(d,e) \in R^{\mathcal{I}_i}$ since we assume $\mathcal{O} \models S \sqsubseteq R$.

If R is rigid, we consider the case that $(d,e) \in R^{\mathcal{I}_j}$ for some $j \neq i$ and get $d, e \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$, as above. Since $R \in \mathsf{N}_{\mathsf{RR}}$, $\mathcal{A}_{\mathsf{R}}$ is an ABox type, and $\mathcal{I}_j \models \mathcal{A}_{\mathsf{R}}$, we must have $R(d,e) \in \mathcal{A}_{\mathsf{R}}$. $\mathcal{I}_i \models \mathcal{A}_{\mathsf{R}}$ then leads to $(d,e) \in R^{\mathcal{I}_i}$. □

The following observation is a direct consequence of the fact that the interpretation of roles in $\mathcal{J}_i$ is based on the canonical interpretations, which are models of $\mathcal{O}$, and Fact 6.10.

Lemma 6.12 *For all $i, j \in [0, n+k]$, elements $d \in \Delta$ and $e \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$, and roles $R \in \mathsf{N}^-_{\mathsf{R}}$, we have:*

$$\text{If } (d,e) \in R^{\mathcal{J}_i} \text{ then } (d,e) \in R^{\mathcal{I}_j}.$$

There are similar connections between the interpretations of concepts in $\mathcal{J}_i$ and $\mathcal{I}_j$.

Lemma 6.13 *For all $i, j \in [0, n+k]$ and basic concepts $B \in \mathbb{B}(\mathcal{O})$, the following hold.*

a) *For all $e \in \mathsf{N_I}(\mathcal{K})$, we have $e \in B^{\mathcal{J}_i}$ iff $e \in B^{\mathcal{I}_i}$.*

b) *If B is rigid, then, for every $e \in \Delta^{\mathcal{I}_j}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{t}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{u}}$, we have $e \in B^{\mathcal{J}_i}$ iff $e \in B^{\mathcal{I}_j}$.*

c) *If B is flexible, then, for every $e \in \Delta^{\mathcal{I}_j}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{t}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{u}}$, we have $e \in B^{\mathcal{J}_i}$ iff*
- *$i = j$ and $e \in B^{\mathcal{I}_i}$, or*
- *there is a $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ with $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq B$, or*
- *$e \in B^{\mathcal{I}_j} \cap \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ and $\mathcal{W}_{\mathcal{O}}(e) \neq \emptyset$.*

Proof. For a), we first assume B to be rigid. ($\Leftarrow$) $e \in B^{\mathcal{I}_i}$ clearly implies $e \in B^{\mathcal{J}_i}$ since $X^{\mathcal{I}_i} \subseteq X^{\mathcal{J}_i}$ for $X \in \mathsf{N_{RC}} \cup \mathsf{N_{RR}}$. ($\Rightarrow$) By the definition of the rigid names in $\mathcal{J}_i$, $e \in B^{\mathcal{J}_i}$ yields that there is a $j \in [0, n+k]$ such that $e \in B^{\mathcal{I}_j}$. Since $\mathcal{I}_j$ and $\mathcal{I}_i$ are both models of the ABox type $\mathcal{A}_{\mathsf{R}}$, we get $B(e) \in \mathcal{A}_{\mathsf{R}}$ and $e \in B^{\mathcal{I}_i}$.

Regarding B being flexible, the definition of $\mathcal{J}_i$ on this kind of names yields two options equivalent to $e \in B^{\mathcal{J}_i}$: (i) $e \in B^{\mathcal{I}_i}$ or (ii) there are a $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ with $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq B$ and a $j \in [0, n+k]$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$. For B being of the form $\exists R$ with $R \in \mathsf{N}^-_{\mathsf{R}}$, this choice is a consequence of the following observations, assuming (i) does not apply:

- If there is an $S \in \mathsf{N}^-_{\mathsf{RR}}$ such that $\mathcal{O} \models S \sqsubseteq R$ and $e \in (\exists S)^{\mathcal{I}_j}$, $\mathcal{O} \models \bigsqcap\{\exists S\} \sqsubseteq \exists R$ holds as well. Hence, we can set $\mathcal{B} := \{\exists S\}$.

- Otherwise we can assume that there is an element $u^j_{eS} \in \Delta$ such that $(e, u^j_{eS}) \in R^{\mathcal{I}_j}$ because of Fact 6.10 and Definition 6.4, which yields that the elements in $\mathsf{N_I}(\mathcal{K}) \cup \Delta^{\mathcal{I}_j}_{\mathsf{a}}$ do not have witnesses; by Definition 2.11, we especially have $\mathcal{O} \models S \sqsubseteq R$. Given $\mathcal{W}_{\mathcal{O}}(e) = \emptyset$, $(e, u^j_{eS}) \in R^{\mathcal{I}_j}$ implies that $\mathcal{W}_{\mathcal{O}}(u^j_{eS}) \neq \emptyset$, meaning that there is a witness $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S$. Additionally, $\mathcal{O} \models \exists S \sqsubseteq \exists R$ implies $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R$.

Regarding now the two options, we get that (ii) implies (i) since $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ yields $B'(e) \in \mathcal{A}_{\mathsf{R}}$ for all $B' \in \mathcal{B}$, which together with $\mathcal{I}_i \models \mathcal{A}_{\mathsf{R}}$ implies $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_i}$. Hence, $\mathcal{I}_i \models \mathcal{O}$ leads to $e \in B^{\mathcal{I}_i}$. This concludes the proof of a).

b) is a direct consequence Fact 6.10 and the definition of $\mathcal{J}_i$.

For c) and the case that $B \in \mathsf{N_C}$, the equivalence with one of the three cases is covered by the definition of $\mathcal{J}_i$ if, for the last case, Lemma 2.15 is taken into account.

It remains to consider B to be of the form $\exists R$ with $R \in \mathsf{N}^-_{\mathsf{R}}$. For the case that $i = j$, the claim can be restricted to the first of the three items since the other two are subsumed by it; for the second item, this holds because $\mathcal{I}_i \models \mathcal{O}$. Then, it is a direct consequence of Fact 6.10 and the definition of $\mathcal{J}_i$, because the interpretation of the elements from $\Delta^{\mathcal{I}_i}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_i}_{\mathsf{t}} \cup \Delta^{\mathcal{I}_i}_{\mathsf{u}}$ in $\mathcal{J}_i$ is not influenced by any $\mathcal{I}_j$ with $j \neq i$. We consider the case $i \neq j$.

- Regarding $e \in \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}} \cup \mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$, we only have to consider the second item. ($\Rightarrow$) The definition of $R^{\mathcal{J}_i}$ yields two options: (i) there is an $S \in \mathsf{N}^-_{\mathsf{RR}}$ such that $e \in (\exists S)^{\mathcal{J}_i}$ and $\mathcal{O} \models S \sqsubseteq R$, or (ii) there is an R-successor d of e in $\mathcal{I}_j$, and either e or d

has a witness w.r.t. $\mathcal{O}$. For (i), we set $\mathcal{B} := \{\exists S\}$. For (ii), Definition 6.4 yields $\mathcal{W}_\mathcal{O}(e) = \emptyset$ and thus that d is of the form u^j_{eS} with $S \in \mathsf{N}^-_\mathsf{R}(\mathcal{O})$ since $\mathcal{W}_\mathcal{O}(d) \neq \emptyset$. The latter implies that there is a $\mathcal{B} \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ with $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S$. Taking $\mathcal{O} \models S \sqsubseteq R$ into account, which follows from Definition 2.11 because of the shape of d, we then get $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R$. ($\Leftarrow$) Given $e \in (\exists R)^{\mathcal{I}_j}$, there is an element of the form $u^j_{eR} \in \Delta^{\mathcal{I}_j}_\mathsf{u}$ such that $(e, u^j_{eR}) \in R^{\mathcal{I}_j}$ by Definition 2.11. Since Definition 6.4 implies that $\mathcal{B}$ is a witness of u^j_{eR}, the definition of $\mathcal{J}_i$ yields $(e, u^j_{eR}) \in R^{\mathcal{J}_i}$; that is, $e \in (\exists R)^{\mathcal{J}_i}$.

- Let $e \in \Delta^{\mathcal{I}_j}_\mathsf{u}$. ($\Rightarrow$) Again, there are two options, by the definition of $\mathcal{J}_i$: (i) there is an $S \in \mathsf{N}^-_{\mathsf{RR}}$ such that $e \in (\exists S)^{\mathcal{I}_j}$ and $\mathcal{O} \models S \sqsubseteq R$, or (ii) there is an R-successor d of e in $\mathcal{I}_j$, and either e or d has a witness w.r.t. $\mathcal{O}$. For (i), we can set $\mathcal{B} := \{\exists S\}$, as in the previous item. For (ii), observe that we have $e \in (\exists R)^{\mathcal{I}_j}$. If $\mathcal{W}_\mathcal{O}(d)$ is undefined or empty, then the third item of c) holds directly. Otherwise, we assume $\mathcal{W}_\mathcal{O}(d) \neq \emptyset$ and hence get $d \in \Delta^{\mathcal{I}_j}_\mathsf{u}$ by Definition 6.4. By Definition 2.11, we then have either (i') $e = u^j_\varrho$ and $d = u^j_{\varrho R}$, or (ii') $d = u^j_\varrho$ and $e = u^j_{\varrho R^-}$. For (i'), Definition 6.4 yields either $\mathcal{W}_\mathcal{O}(e) \neq \emptyset$ (i.e., the third item holds) or that there is a $\mathcal{B} \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R$ (the second item).

  For (ii'), we immediately get that the witness of d is also a witness of e, again by Definition 6.4. ($\Leftarrow$) Let $e = u^j_\varrho$. We start with the second item. If there is a set $\mathcal{B} \in \mathbb{B}_\mathsf{R}(\mathcal{O})$ with $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R$, then Definition 2.11 implies that the element $u^j_{\varrho R} \in \Delta^{\mathcal{I}_j}_\mathsf{u}$ exists and that $(e, u^j_{\varrho R}) \in R^{\mathcal{I}_j}$. Since $\mathcal{B}$ is further a witness of $u^j_{\varrho R}$ by Definition 6.4, we get $(e, u^j_{\varrho R}) \in R^{\mathcal{J}_i}$, and hence $e \in (\exists R)^{\mathcal{J}_i}$ by the definition of $\mathcal{J}_i$. Regarding item three, and thus $e \in (\exists R)^{\mathcal{I}_j}$, we similarly get that $(e, u^j_{\varrho R}) \in R^{\mathcal{I}_j}$ by Definition 2.11. Then, $\mathcal{W}_\mathcal{O}(e) \neq \emptyset$ yields $e \in (\exists R)^{\mathcal{J}_i}$, as in the previous case. □

We finally show that $\mathcal{J}_i$ is in fact as intended. The proof of the next lemma is standard given Lemmas 6.11 and 6.13.

Lemma 6.14 *For all $i \in [0, n+k]$, $\mathcal{J}_i$ is a model of $(\mathcal{O}, \mathcal{A}_i)$.*

Proof. For every assertion $\alpha \in \mathcal{A}_i$, Lemma 2.12 yields $\mathcal{I}_i \models \alpha$, and thus Lemmas 6.11 and 6.13a) yield $\mathcal{J}_i \models \alpha$.

We consider a CI $B_1 \sqcap \cdots \sqcap B_m \sqsubseteq B \in \mathcal{O}$ and element e such that $e \in B_1^{\mathcal{J}_i} \cap \cdots \cap B_m^{\mathcal{J}_i}$; note that $B_1, \ldots, B_m$ are basic concepts and B is either a basic concept or $\bot$. For the case that $e \in \mathsf{N_I}(\mathcal{K})$, Lemma 6.13a) yields $e \in B_1^{\mathcal{I}_i} \cap \cdots \cap B_m^{\mathcal{I}_i}$. Since $\mathcal{I}_i \models \mathcal{O}$, this implies that $e \in B^{\mathcal{I}_i}$, which is impossible if $B = \bot$. Otherwise, we get $e \in B^{\mathcal{J}_i}$, again by Lemma 6.13a).

Let now $e \in \Delta^{\mathcal{I}_j}_\mathsf{a} \cup \Delta^{\mathcal{I}_j}_\mathsf{t} \cup \Delta^{\mathcal{I}_j}_\mathsf{u}$ for some $j \in [0, n+k]$. If $i = j$, then we get the same conclusion as in the previous case since, given that $\mathcal{I}_j \models \mathcal{O}$, items two and three collapse to the first item. More precisely, we can argument analogously by referring to Lemma 6.13b) and c) instead of Lemma 6.13a). Regarding $i \neq j$, Lemma 6.13 implies that, for each B_ℓ with $\ell \in [1, m]$, we have either that (i) there is a $\mathcal{B}_\ell \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ such that $e \in (\bigsqcap \mathcal{B}_\ell)^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B}_\ell \sqsubseteq B_\ell$ or that (ii) $e \in B_\ell^{\mathcal{I}_j} \cap \Delta^{\mathcal{I}_j}_\mathsf{u}$ and $\mathcal{W}_\mathcal{O}(e) \neq \emptyset$. $\mathcal{I}_j \models \mathcal{O}$, following from Lemma 2.12, thus leads to $e \in B_\ell^{\mathcal{I}_j}$ for (i) and to $e \in B^{\mathcal{I}_j}$ for both cases. If $B = \bot$, this is again impossible. If B is rigid, then Lemma 6.13b) yields $e \in B^{\mathcal{J}_i}$, as

required. If B is flexible and Case (ii) applies to at least one B_ℓ with $\ell \in [1, m]$, then the third item of Lemma 6.13c) yields the claim. Otherwise, it is easy to see that we can define $\mathcal{B} := \bigcup_{\ell=1}^{m} \mathcal{B}_\ell$ and have $e \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$, $\mathcal{O} \models \mathcal{B} \sqsubseteq B_1 \sqcap \cdots \sqcap B_m$, and $\mathcal{O} \models \mathcal{B} \sqsubseteq B$. Hence, the second item of Lemma 6.13c) applies, and we also get $e \in B^{\mathcal{J}_i}$.

It remains to consider role inclusions of the form $S \sqsubseteq R$. We consider a tuple $(d, e) \in S^{\mathcal{J}_i}$ and focus on the definition of $\mathcal{J}_i$ regarding roles, in the following. That is, there is a $j \in [0, n + k]$ such that $(d, e) \in S^{\mathcal{I}_j}$. Since $\mathcal{I}_j \models \mathcal{O}$, we get $(d, e) \in R^{\mathcal{I}_j}$. For the case that R is rigid, we immediately get $(d, e) \in R^{\mathcal{J}_i}$. For the case that R is flexible, we assume $(d, e) \notin R^{\mathcal{J}_i}$, by contradiction. Then, we must have that $i \neq j$, that R has no rigid subrole S' such that $(d, e) \in S'^{\mathcal{I}_j}$, and that neither d nor e have a witness w.r.t. $\mathcal{O}$. The second of these observations implies that S is neither rigid and has no rigid subrole S' such that $(d, e) \in (S')^{\mathcal{I}_j}$, because S would also be a rigid subrole of R. But this means $(d, e) \notin S^{\mathcal{I}_j}$, which contradicts our assumption. □

We now provide the final missing piece to show r-satisfiability of $\mathcal{W}$ and prove that, for all $i \in [0, n + k]$, the interpretation $\mathcal{J}_i$ satisfies the corresponding conjunction χ_i of CQ literals. Regarding the positive literals, this is easy given that the ABox $\mathcal{A}_{Q_{\iota(i)}}$ contains an instantiation of all these CQs and is satisfied by $\mathcal{I}_i$. The proof for the negative literals $\neg\varphi$ is based on the interpretation of roles in $\mathcal{J}_i$, where domain elements of different interpretations $\mathcal{I}_j$ and $\mathcal{I}_\ell$ with $j, \ell \in [0, n + k]$ are only related sparsely (i.e., at least one of two such elements must be a named individual). Given that we assume CQs to be connected, we specifically have that, if $\mathcal{J}_i$ satisfies φ, one must apply: (I) the corresponding homomorphism includes only unnamed domain elements of a single $\mathcal{I}_j$ with $j \in [0, n + k]$, and a witness query of φ is satisfied in $\mathcal{I}_j$, which follows from the structure of the unnamed domain parts, or (II) it includes named elements, and either it maps directly into $\mathcal{I}_i$ or we can construct such a homomorphism; the latter holds because the given homomorphism contains named domain elements from some $\mathcal{I}_j$ with $j \in [0, n + k]$ and $j \neq i$, and corresponding rigid knowledge on the named individuals must be contained in the additional ABoxes and thus also be satisfied in $\mathcal{I}_i$. Based on the assumed r-completeness, we therefore can show $\mathcal{J}_i \models \neg\varphi$ by contradiction.

Lemma 6.15 *For all $i \in [0, n + k]$, $\mathcal{J}_i$ is a model of χ_i.*

Proof. We show that $\mathcal{J}_i$ is a model of every CQ literal in χ_i. Let φ first be a positive such literal; note that the proof of this case is analogous to the proof for $\mathcal{EL}$ (see Lemma 5.13). Since $\mathcal{A}_{Q_i}$ contains an instantiation of φ and $\mathcal{I}_i \models \mathcal{A}_{Q_i}$ by Lemma 2.12[7] we know that there is a homomorphism π of φ into $\mathcal{I}_i$ that maps all variables in φ to elements of $\Delta^{\mathcal{I}_i}_{\mathsf{a}}$; that is, π maps each such variable x to a_x. By the fact that $\Delta^{\mathcal{I}_i}_{\mathsf{a}} \subseteq \Delta$ and Lemmas 6.11 and 6.13, π is then also a homomorphism of φ into $\mathcal{J}_i$.

Let now $\neg\varphi$ be a negative literal in χ_i. We proceed by contradiction and assume π to be a homomorphism of φ into $\mathcal{J}_i$. First, observe the following:

- Condition (C4) implies $\varphi \in Q^{\neg}_{\mathsf{R}}$, and hence Condition (C5) yields that none of the witnesses of φ is satisfied in any of the canonical interpretations.
- Condition (C2) implies $\mathcal{K}^i_{\mathsf{R}} \not\models \varphi$, and thus Lemma 2.13 yields that $\mathcal{I}_i \models \neg\varphi$.

[7] Note that, in the remaining parts of the proof, we do not always explicitly refer to Lemma 2.12 to justify the argument that $\mathcal{I}_i \models \mathcal{K}^i_{\mathsf{R}}$ for all $i \in [0, n + k]$.

We derive contradictions to these observations by distinguishing two cases (I) and (II) outlined above.

(I) Let first π be such that it maps no terms into $\mathsf{N_I}(\mathcal{K}) \cup \bigcup_{j=0}^{n+k} \Delta_{\mathsf{a}}^{\mathcal{I}_j} \cup \Delta_{\mathsf{t}}^{\mathcal{I}_j}$. Because of the UNA, we thus have $\mathsf{N_I}(\varphi) \cap \mathsf{N_I}(\mathcal{K}) = \emptyset$, which yields that $\mathsf{N_I}(\varphi) = \emptyset$ since φ does not contain any names from $\mathsf{N_I^{aux}}$ or $\mathsf{N_I^{tree}}$. Moreover, we can then assume that there is a single index $j \in [0, n+k]$ such that π maps all terms of φ to elements of $\Delta_{\mathsf{u}}^{\mathcal{I}_j}$, by the definition of Δ. To see this, note that φ is connected and that, for a role $R \in \mathsf{N_R^-}$, a tuple $(d, e) \in R^{\mathcal{J}_i}$ without named individuals exists only if the elements belong to the same domain $\Delta^{\mathcal{I}_j}$ by the definition of $R^{\mathcal{J}_i}$ and Fact 6.10.

Given the above observation that $\mathcal{I}_i \models \neg\varphi$, we then directly get a contradiction for the case that $j = i$ by Lemmas 6.11 and 6.13, which together imply that π is a homomorphism of φ into $\mathcal{I}_i$.

For the case $j \neq i$, we show that there is a witness query ψ for φ such that $\mathcal{I}_j \models \psi$, which implies $\mathcal{I}_j \models \varphi$ by Lemma 6.6. Since $\mathcal{I}_j \models \neg\varphi$ is a consequence of the above observation that $\varphi \in Q_{\mathsf{R}}^{\neg}$, Condition (C5), and Lemma 2.13, we get a contradiction. In the remains of Part (I), we construct this witness query. We start defining a tree witness f for φ, then consider an element of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ as required according to Definition 6.5 for both kinds of witness queries, and lastly define ψ.

By the fact that φ is connected, by Lemma 6.12, and by considering how the elements in $\Delta_{\mathsf{u}}^{\mathcal{I}_j}$ are related by roles within $\mathcal{I}_j$ (see Definition 2.11), it is easy to see that there is a variable $x \in \mathsf{N_V}(\varphi)$, for which $\pi(x) = u_{\varrho}^j$ is such that the length of ϱ is minimal compared to the paths of all elements of $\mathsf{range}(\pi)$; further, all other $\pi(y)$ for $y \in \mathsf{N_V}(\varphi)$ must then be of the form $u_{\varrho\sigma}^j$ with $\varrho\sigma \in (\mathsf{N_R^-})^*$. We define f as a tree witness f_x w.r.t. x and via a mapping $\mathrm{f}' \colon \mathsf{range}(\pi) \to (\mathsf{N_R^-} \times 2^{\mathsf{N_R^-}})^*$, by setting $\mathrm{f}(y) := \mathrm{f}'(\pi(y))$ for all $y \in \mathsf{N_V}(\varphi)$. Regarding f', we set $\mathrm{f}'(u_{\varrho}^j) := \epsilon$ and proceed by induction over the structure of $\Delta_{\mathsf{u}}^{\mathcal{I}_j}$. So we consider an element $u_{\varrho\sigma S_1}^j \in \mathsf{range}(\pi)$ such that $\mathrm{f}'(u_{\varrho\sigma}^j)$ has already been defined. We set $\mathrm{f}'(u_{\varrho\sigma S_1}^j) := \mathrm{f}'(u_{\varrho\sigma}^j) \cdot (S_1, \mathcal{R})$, where $\mathcal{R}$ is a minimal set satisfying the following conditions for all role atoms $R(y, z) \in \varphi$ with $\pi(y) = u_{\varrho\sigma}^j$ and $\pi(z) = u_{\varrho\sigma S_1}^j$:

- If there is an $S \in \mathsf{N_{RR}^-}(\mathcal{O})$ such that $\mathcal{O} \models S \sqsubseteq R$ and $(\pi(y), \pi(z)) \in S^{\mathcal{J}_i}$, then $S \in \mathcal{R}$.
- Otherwise, we have $R \in \mathcal{R}$.

Regarding the second requirement of Definition 6.3, observe the following. From the assumption that $(\pi(y), \pi(z)) \in R^{\mathcal{J}_i}$ and, if the first item applies, $(\pi(y), \pi(z)) \in S^{\mathcal{J}_i}$, we get $(\pi(y), \pi(z)) \in R^{\mathcal{I}_j}$ and, respectively, $(\pi(y), \pi(z)) \in S^{\mathcal{I}_j}$ by Lemma 6.12. Definition 2.11 then implies $\mathcal{O} \models S_1 \sqsubseteq R$ and $\mathcal{O} \models S_1 \sqsubseteq S$, respectively. Hence, f is indeed a tree witness for x in φ.

Next, we construct an element of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$. Note that the last two properties in the definition of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ do not depend on the particular choice of that set, but only on f (see Definition 6.3). We first show that they are fulfilled and, to this end, consider all atoms on variables y with $\mathrm{f}(y) \neq \epsilon$. Let $A(y) \in \varphi$ and $\mathrm{f}(y) = \sigma \cdot (S, \mathcal{R})$, which implies $\pi(y) \in A^{\mathcal{J}_i}$ and that $\pi(y)$ is of the form $u_{\varrho\sigma|_1 S}^j$; $\sigma\,|_1$ denotes the sequence of roles obtained from σ by projecting each pair in σ onto its first component. Then, all options we get by applying Lemma 6.13 yield $u_{\varrho\sigma|_1 S}^j \in A^{\mathcal{I}_j}$, since $\mathcal{I}_j \models \mathcal{O}$. Hence, $\mathcal{O} \models \exists S^- \sqsubseteq A$ follows

by Lemma 2.15. Second, we consider an element $\sigma \cdot (S_1, \mathcal{R}_1) \cdot (S_2, \mathcal{R}_2) \in \mathsf{range}(\mathrm{f})$. By Definition 2.11, we know that $(u^j_{\varrho\sigma|_1 S_1}, u^j_{\varrho\sigma|_1 S_1 S_2}) \in S_2^{\mathcal{I}_j}$, and hence get $u^j_{\varrho\sigma|_1 S_1} \in (\exists S_2)^{\mathcal{I}_j}$. Again by Lemma 2.15, we obtain $\mathcal{O} \models \exists S_1^- \sqsubseteq \exists S_2$.

For the actual construction of the element of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ and the corresponding witness query, we distinguish two cases focusing on the two kinds of witness queries, respectively (see Definition 6.5).

- If $\mathcal{W}_{\mathcal{O}}(u^j_\varrho) = \emptyset$, then we construct a tree witness query. We continue the construction of a set $\mathcal{B} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ and thereby ensure that the first two conditions in the definition of $\mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ are satisfied. Let ϱ be of the form $\varrho_0 S_0$. For each atom $\alpha := A(y)$ ($\alpha := R(y, z)$) in φ such that $\pi(y) = u^j_\varrho$ (i.e., $\mathrm{f}(y) = \epsilon$), we regard the basic concept $B_\alpha := A$ ($B_\alpha := \exists S_1$, assuming $\pi(z) = u^j_{\varrho S_1}$). We then define a set $\mathcal{B}_\alpha \subseteq \mathbb{B}$ as follows: If there is a set $\mathcal{B}'_\alpha \subseteq \mathbb{B}_{\mathsf{R}}$ such that $\mathcal{O} \models \exists S_0^- \sqsubseteq \bigsqcap \mathcal{B}'_\alpha$ and $\mathcal{O} \models \bigsqcap \mathcal{B}'_\alpha \sqsubseteq B_\alpha$, then $\mathcal{B}_\alpha := \mathcal{B}'_\alpha$ and, otherwise, $\mathcal{B}_\alpha := \{B_\alpha\}$. Note that the first case always applies if α is a concept atom $A(y)$, by Lemma 6.13, because $u^j_\varrho \in A^{\mathcal{J}_i}$, $\mathcal{W}_{\mathcal{O}}(u^j_\varrho) = \emptyset$, and $A = B_{A(y)}$. By setting $\mathcal{B} := \bigcup_{\alpha \in \varphi} \mathcal{B}_\alpha$, we thus get $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq A$ ($\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S_1$).

  We then also have $u^j_\varrho \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ since $u^j_\varrho \in (\bigsqcap \mathcal{B}_\alpha)^{\mathcal{I}_j}$ holds for all $\alpha \in \varphi$. This can be seen by considering the definition of $\mathcal{B}_\alpha$. Regarding the first case, the latter follows from $\varrho = \varrho_0 S_0$ and $\exists S_0^- \sqsubseteq \bigsqcap \mathcal{B}_\alpha$ by Lemma 2.15. Regarding the second case, we can consider some $\alpha := R(y, z)$ and $\mathcal{B}_\alpha := \{\exists S_1\}$, which yields that $\pi(z) = u^j_{\varrho S_1} \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ and thus $(u^j_\varrho, u^j_{\varrho S_1}) \in S_1^{\mathcal{I}_j}$ by Definition 2.11. This means $u^j_\varrho \in (\bigsqcap\{\exists S_1\})^{\mathcal{I}_j}$. It can then readily be checked that $\mathcal{B} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$. In particular, observe that there is such a role atom $R(y, z) \in \varphi$ where $\mathrm{f}(z) = (S_1, \mathcal{R})$, for each $(S_1, \mathcal{R}) \in \mathsf{range}(\mathrm{f})$. This is because the latter implies that φ contains a variable z such that $\pi(z) = u^j_{\varrho S_1}$ by the definition of f, we assume $\pi(y) = u^j_\varrho$, φ including y and z is connected, and, by Lemma 6.12, $u^j_{\varrho S_1}$ and u^j_ϱ are equally connected in $\mathcal{I}_j$; but this is only possible if they are related by a role. Recall that, if we consider $\mathrm{f}(z) = \sigma \cdot (S_1, \mathcal{R})$ as in the latter sentence, we can assume $\pi(z)$ to be of the form $u^j_{\varrho\sigma|_1 S_1}$ by our definition of f.

  Based on $\mathcal{B}$, we now define a tree witness query ψ (see Definition 6.5) as required. To this end, we maintain the invariant that $\mathcal{I}_j \models \psi$ during the construction, via the homomorphism that maps all variables x_σ with $\sigma \in \mathsf{range}(\mathrm{f})$ to $u^j_{\varrho\sigma|_1}$. We proceed by induction over $\mathsf{range}(\mathrm{f})$. The start is by including into ψ all atoms from $\mathcal{B}|_{\mathsf{R}}(x_\epsilon)$. Hence, both the first condition of Definition 6.5 and our invariant are satisfied since we showed $u^j_\varrho \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ above. The second condition is satisfied since, for all $A(y) \in \varphi$ with $\mathrm{f}(y) = \epsilon$, we have $\mathcal{B}_{A(y)} \subseteq \mathcal{B} \cap \mathbb{B}_{\mathsf{R}}$, $\mathcal{O} \models \bigsqcap \mathcal{B}_{A(y)} \sqsubseteq B_{A(y)}$, and $B_{A(y)} = A$. Condition three is trivially fulfilled in the base case. Regarding the last condition we consider an atom $R(y, z) \in \varphi$ with $\mathrm{f}(y) = \epsilon$ and $\mathrm{f}(z) = (S_1, \mathcal{R})$. For those $(S_1, \mathcal{R}) \in \mathsf{range}(\mathrm{f})$ for which $\mathcal{B}_{R(y,z)}$ is defined in the first case of that definition (i.e., we have $\mathcal{B}_{R(y,z)} \subseteq \mathbb{B}_{\mathsf{R}}$ and $\mathcal{O} \models \bigsqcap \mathcal{B}_{R(y,z)} \sqsubseteq B_{R(y,z)}$) $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S_1$ follows directly from $\mathcal{B}_{R(y,z)} \subseteq \mathcal{B}$ and $B_{R(y,z)} = \exists S_1$. Hence, $(S_1, \mathcal{R})$ is rigidly witnessed in ψ by Definition 6.5. For all other $(S_1, \mathcal{R}) \in \mathsf{range}(\mathrm{f})$, we know that $(S_1, \mathcal{R})$ cannot be rigidly witnessed in ψ. Observe that we then have $\mathcal{R} \subseteq \mathsf{N}_{\mathsf{RR}}$: if there was a flexible role in $\mathcal{R}$, then it would have to be introduced for an atom of the

form $R(y,z)$ that was flexible and to equal R, and there could not be an $S \in \mathsf{N_{RR}}$ such that $\mathcal{O} \models S \sqsubseteq R$ and $(\pi(y),\pi(z)) \in S^{\mathcal{J}_i}$ by the construction of $\mathcal{R}$; then, there could neither be an $S \in \mathsf{N_{RR}}$ such that $\mathcal{O} \models S \sqsubseteq R$ and $(\pi(y),\pi(z)) \in S^{\mathcal{I}_j}$ since the definition of $\mathcal{J}_i$ would yield $(\pi(y),\pi(z)) \in S^{\mathcal{J}_i}$, which contradicts the former assumption; by the definition of $R^{\mathcal{J}_i}$, $(\pi(y),\pi(z)) \in R^{\mathcal{J}_i}$, $\pi(y) = u^j_\varrho$, $\pi(z) = u^j_{\varrho S_1}$, and $\mathcal{W}_\mathcal{O}(u^j_\varrho) = \emptyset$ would thus imply $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) \neq \emptyset$; but, according to Definition 6.4, the latter means that there is a set $\mathcal{B}'_{R(y,z)} \subseteq \mathbb{B}_\mathsf{R}$ such that $\mathcal{O} \models \bigsqcap \mathcal{B}'_{R(y,z)} \sqsubseteq \exists S_1$ and $u^j_\varrho \in (\bigsqcap \mathcal{B}'_{R(y,z)})^{\mathcal{I}_j}$, yielding $\mathcal{O} \models \exists S_0^- \bigsqcap \mathcal{B}'_{R(y,z)}$ by Lemma 2.15, which contradicts the assumption that such a set does not exist (i.e., since $\mathcal{B}_{R(y,z)}$ is not defined in the first case of the definition). In order to fulfill the last condition, we thus can add all atoms $S(x_\epsilon, x_{(S_1,\mathcal{R})})$, $S \in \mathcal{R}$ to ψ. Since Definition 2.11 implies $(u^j_\varrho, u^j_{\varrho S_1}) \in S_1^{\mathcal{I}_j}$ and we have $\mathcal{O} \models S_1 \sqsubseteq S$ for all these roles S (see the part below the construction of $\mathcal{R}$), Definition 2.11 also implies $(u^j_\varrho, u^j_{\varrho S_1}) \in S^{\mathcal{I}_j}$. Hence, these new atoms can be mapped into $\mathcal{I}_j$, as required for our invariant.

For the induction step, conditions three and four of Definition 6.5 are relevant. Since the elements $(S_1,\mathcal{R}) \in \mathsf{range}(\mathrm{f})$ considered lastly are not rigidly witnessed, we now focus on all of $\mathsf{range}(\mathrm{f})$ which extend those. To this end, we assume that we have already defined ψ up to a variable of the form x_σ, in line with Definition 6.5, and that there is some $\sigma \cdot (S_2,\mathcal{R}) \in \mathsf{range}(\mathrm{f})$ that is not rigidly witnessed. Regarding condition three, we consider all $A(y) \in \varphi$ of the form $\mathrm{f}(y) = \sigma \cdot (S_2,\mathcal{R})$. Since $\pi(y) \in A^{\mathcal{J}_i}$, Lemma 6.13 yields that either (i') there is a set $\mathcal{B}_A \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B}_A \sqsubseteq A$ and $\pi(y) \in (\bigsqcap \mathcal{B}_A)^{\mathcal{I}_j}$, or (ii') $\pi(y) \in A^{\mathcal{I}_j}$ and $\mathcal{W}_\mathcal{O}(\pi(y)) \neq \emptyset$. In Case (i'), we add the atoms $\mathcal{B}_A(y_{\sigma\cdot(S_2,\mathcal{R})})$ to ψ and thus satisfy condition three while maintaining our invariant. Case (ii') cannot apply since it leads to a contradiction as follows. For $\mathcal{W}_\mathcal{O}(\pi(y)) \neq \emptyset$, Definition 6.4 yields that there must be a prefix $\varrho' S'$ of $\varrho\sigma|_1 S_2$ and a set $\mathcal{B}_{S'} \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B}_{S'} \sqsubseteq \exists S'$ and either $u^j_{\varrho'} \in (\bigsqcap \mathcal{B}_{S'})^{\mathcal{I}_j}$ or $\varrho' \in \mathsf{N_I}(\mathcal{K}) \cup \mathsf{N_I^{aux}}$ and $\varrho' \in (\bigsqcap \mathcal{B}_{S'})^{\mathcal{I}_j}$. If $\varrho' S'$ is a prefix of ϱ, then this contradicts the assumption that $\mathcal{W}_\mathcal{O}(u^j_\varrho) = \emptyset$. But, if ϱ is a prefix of $\varrho' S'$, then there is a $\varrho'' \in \mathsf{range}(\mathrm{f})$ such that $\varrho'' \mid_1 = \varrho'$, and we have added the atoms in $\mathcal{B}_{S'}(y_{\varrho''})$ to ψ already. But then $\sigma \cdot (S_2,\mathcal{R})$ is rigidly witnessed in ψ, which contradicts the assumption.

Regarding the fourth condition, we have to consider all elements of the form $\sigma \cdot (S_2,\mathcal{R}_2) \cdot (S_3,\mathcal{R}_3) \in \mathsf{range}(\mathrm{f})$. We discern two cases, similar to the above definition of $\mathcal{B}'_\alpha$ for $\mathcal{B}$. If there is a set $\mathcal{B}_{\exists S_3} \subseteq \mathbb{B}_\mathsf{R}$ such that $\mathcal{O} \models \exists S_2^- \sqsubseteq \bigsqcap \mathcal{B}_{\exists S_3}$ and $\mathcal{O} \models \bigsqcap \mathcal{B}_{\exists S_3} \sqsubseteq \exists S_3$, then we add the atoms $\mathcal{B}_{\exists S_3}(x_{\varrho\sigma\cdot(S_2,\mathcal{R}_2)})$ to ψ and get that $\sigma \cdot (S_2,\mathcal{R}_2) \cdot (S_3,\mathcal{R}_3)$ is rigidly witnessed in ψ. Otherwise, we have $\mathcal{R}_2 \subseteq \mathsf{N_{RR}}$ for the same reasons as in the base case. We hence can add all the atoms $S(x_{\sigma\cdot(S_2,\mathcal{R}_2)}, x_{\sigma\cdot(S_2,\mathcal{R}_2)\cdot(S_3,\mathcal{R}_3)})$, $S \in \mathcal{R}_2$, to ψ and continue the inductive construction with $\sigma \cdot (S_2,\mathcal{R}_2) \cdot (S_3,\mathcal{R}_3)$, which is not rigidly witnessed.

It is easy to see that this construction of ψ terminates since f is finite. Moreover, the final ψ satisfies Definition 6.5 and is hence a tree witness query for φ w.r.t. $\mathcal{B}$ and f; and we have that $\mathcal{I}_j \models \psi$.

- If $\mathcal{W}_\mathcal{O}(u^j_\varrho) \neq \emptyset$, then we construct a witness query of the second kind. By Definition 6.4, ϱ is of the form $\varrho = \varrho_0 R_0 \dots R_\ell$, and there are a set $\mathcal{B} \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ such

that $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R_0$ and an element $u^j_{\varrho_0} \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ that satisfies $\bigsqcap \mathcal{B}$ in $\mathcal{I}_j$. By Definition 2.11 and Lemma 2.15, $u^j_{\varrho} \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ leads to $\mathcal{O} \models \exists R_i^- \sqsubseteq \exists R_{i+1}$, $i \in [0, \ell - 1]$, which means that $\mathcal{B}$ is a witness of $\exists R_\ell^-$ w.r.t. $\mathcal{O}$.

It remains to show that $\{\exists R_\ell^-\} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$. But for all $A(y) \in \varphi$ with $\mathrm{f}(y) = \epsilon$, $\pi(y) = u^j_{\varrho} \in A^{\mathcal{J}_i}$ implies $u^j_{\varrho} \in A^{\mathcal{I}_j}$ since $\mathcal{I}_j \models \mathcal{O}$ by Lemma 6.13. Then, Lemma 2.15 yields $\mathcal{O} \models \exists R_\ell^- \sqsubseteq A$. For an element $(S_1, \mathcal{R}) \in (\mathsf{N}_{\mathsf{R}}^- \times 2^{\mathsf{N}_{\mathsf{R}}^-}) \cap \mathsf{range}(\mathrm{f})$, our definition of f similarly implies that the element $u^j_{\varrho S_1} \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ exists. Then, Definition 2.11 leads to $u^j_{\varrho} \in (\exists S_1)^{\mathcal{I}_j}$, and Lemma 2.15 yields $\mathcal{O} \models \exists R_\ell^- \sqsubseteq \exists S_1$. Hence, $\psi := \exists x.\mathcal{B}(x)$ is a witness query for φ and, by regarding the homomorphism that maps x to $u^j_{\varrho_0}$, we get $\mathcal{I}_j \models \psi$.

Given the above observations, this finishes the proof of Case (I) of Lemma 6.15.

(II) In the remainder of the proof, let π be such that it maps at least one term into $\Delta_{\mathsf{n}} := \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \bigcup_{j=0}^{n+k} \Delta^{\mathcal{I}_j}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{t}}$. We directly define a homomorphism π' of φ into $\mathcal{I}_i$ to contradict $\mathcal{I}_i \models \neg\varphi$. This is done in three phases, by considering the terms π maps to elements from Δ_{n}, those that are directly connected to the latter, and all others. After phase one, all terms considered are thus mapped to elements from $\bigcup_{j=0}^{n+k} \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ by π.

1) For all $t \in \mathsf{N}_{\mathsf{I}}(\varphi) \cup \mathsf{N}_{\mathsf{V}}(\varphi)$, where $\pi(t) \in \Delta_{\mathsf{n}}$, and assuming $\pi(t) = e^j$, let $\pi'(t) := e^i$. We first prove an auxiliary result and subsequently show that, regarding the terms mapped so far, π' is a homomorphism of φ into $\mathcal{I}_i$.

Corollary 6.16 *For all $B \in \mathbb{B}(\mathcal{O})$ and $t \in \mathsf{N}_{\mathsf{I}}(\varphi) \cup \mathsf{N}_{\mathsf{V}}(\varphi)$ with $\pi(t) \in \Delta_{\mathsf{n}}$, we have:*

$$\textit{If } \pi(t) \in B^{\mathcal{J}_i}\textit{, then } \pi'(t) \in B^{\mathcal{I}_i}.$$

By Lemma 6.13, $\pi(t) \in B^{\mathcal{J}_i}$ implies two options: (i) $\pi(t) \in B^{\mathcal{I}_i}$, or (ii) $\pi(t) \in \Delta^{\mathcal{I}_j}_{\mathsf{a}} \cup \Delta^{\mathcal{I}_j}_{\mathsf{t}}$ and there is a set $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ such that $\pi(t) \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq B$. In Case (i), we have $\pi'(t) = \pi(t)$ by definition, hence the claim holds. In Case (ii), the claim follows if $i = j$, as in Case (i); otherwise, we distinguish the following two cases.

- If $\pi(t) \in \Delta^{\mathcal{I}_j}_{\mathsf{a}}$, then $\pi(t)$ is of the form a^j_x. Let $\varphi' \in \mathcal{Q}_\Phi$ be the (unique) CQ containing the variable x. By the definition of $\mathcal{I}_j$ and Condition (C3), the existence of the element a^j_x implies that $\varphi' \in Q_{\mathsf{R}}$. Hence, the element a^i_x must also exist, and $\pi'(t) = a^i_x$ is well-defined. Since $\pi(t) \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$; $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$; and $\mathcal{A}_{Q_{\mathsf{R}}}$ and $\mathcal{A}_{R_{\mathsf{F}}}$ contain all rigid assertions on a_x, taking $\mathcal{O}$ into account (i.e., in particular, all following from $\mathcal{A}_{Q_{\iota(j)}}$ are also in $\mathcal{A}_{Q_{\mathsf{R}}}$); Definition 2.11 and Lemma 2.14 yield that the elements of $\mathcal{B}$ are implied by the conjunction of
  - all elements of $\mathbb{B}(\varphi, a_x)$ and
  - all rigid concepts $\exists R$ for which there is an assertion $\exists S(a_x) \in R_{\mathsf{F}}$ with $\mathcal{O} \models S \sqsubseteq R$.

  But, for all concepts of the latter form, Condition (C6) yields that $\exists R(a_x)$ is (already) implied by some KB $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i')}} \cup \mathcal{A}_{i'} \rangle$, and must hence be contained in $\mathbb{B}(\varphi', a_x)$, because $\mathcal{A}_{\mathsf{R}}$ and $\mathcal{A}_{i'}$ do not contain assertions on a_x. This shows that $\mathcal{O} \models \bigsqcap \mathbb{B}(\varphi', a_x) \sqsubseteq \bigsqcap \mathcal{B}$. Given $\varphi' \in Q_{\mathsf{R}}$, the definition of $\mathcal{A}_{Q_{\mathsf{R}}}$ then implies $B'(a_x) \in \mathcal{A}_{Q_{\mathsf{R}}}$ for all $B' \in \mathcal{B}$. From $\mathcal{I}_i \models \mathcal{A}_{Q_{\mathsf{R}}}$, we obtain $a^i_x \in (\bigsqcap \mathcal{B})^{\mathcal{I}_i}$ which together with $\mathcal{I}_i \models \mathcal{O}$ leads to $a^i_x \in B^{\mathcal{I}_i}$, as required.

- If $\pi(t) \in \Delta_{\mathsf{t}}^{\mathcal{I}_j}$, then $\pi(t)$ is of the form $a^j_{b\varrho}$. Since $a_{b\varrho}$ then occurs in some ABox by Definition 2.11, only $\mathcal{A}_{R_\mathsf{F}}$ contains assertions on elements of $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$, and $\mathcal{A}_{R_\mathsf{F}}$ is the same w.r.t. all time points, the element $a^i_{b\varrho}$ also exists. By Lemma 2.14 and the definition of $\mathcal{A}_{R_\mathsf{F}}$, $\pi(t) \in B^{\mathcal{I}_j}$ implies that B subsumes the conjunction of all rigid basic concepts satisfied by $u_{b\varrho}$ in some $\mathcal{I}_{\langle\mathcal{O},\{\exists S(b)\}\rangle}$ where $\exists S(b) \in R_\mathsf{F}$. Another application of Lemma 2.14 then yields that B is also satisfied by $\pi'(t) = a^i_{b\varrho}$ in $\mathcal{I}_i$.

This concludes the proof of Corollary 6.16.

As a consequence, we have that all concept atoms $A(t)$ in φ such that $\pi(t) \in \Delta_\mathsf{n}$ are satisfied by π' in $\mathcal{I}_i$. We next show that this also holds for the role atoms that only contain such terms. Let hence $R(t, t') \in \varphi$ be such that $\pi(t), \pi(t') \in \Delta_\mathsf{n}$. If $\pi(t)$ and $\pi(t')$ are both contained in $\Delta^{\mathcal{I}_i}$, which especially holds for the elements in $\mathsf{N}_\mathsf{I}(\mathcal{K})$, the claim follows immediately from Lemma 6.11 and the fact that $\pi'(t) = \pi(t)$ and $\pi'(t') = \pi(t')$. Otherwise, we have that both $\pi(t)$ and $\pi(t')$ belong to some $\mathsf{N}_\mathsf{I}(\mathcal{K}) \cup \Delta_{\mathsf{a}}^{\mathcal{I}_j} \cup \Delta_{\mathsf{t}}^{\mathcal{I}_j}$ for a fixed $j \in [0, n+k]$ such that $j \neq i$; note that there are no role connections between elements of different sets $\Delta_{\mathsf{a}}^{\mathcal{I}_j} \cup \Delta_{\mathsf{t}}^{\mathcal{I}_j}$ and $\Delta_{\mathsf{a}}^{\mathcal{I}_{j'}} \cup \Delta_{\mathsf{t}}^{\mathcal{I}_{j'}}$ in $\mathcal{J}_i$ by definition. We then can discern the following cases and argue based on our special ABoxes:

- R is rigid, $\pi(t)$ or $\pi(t')$ is contained in $\Delta_{\mathsf{a}}^{\mathcal{I}_j}$, and none is contained in $\Delta_{\mathsf{t}}^{\mathcal{I}_j}$:
  $(\pi(t), \pi(t')) \in R^{\mathcal{J}_i}$ implies $(\pi(t), \pi(t')) \in R^{\mathcal{I}_j}$. By Definition 2.11, specifically regarding relations between named individuals, and the fact that $\mathcal{A}_\mathsf{R}$ and $\mathcal{K}$ do not contain assertions on elements from $\mathsf{N}_\mathsf{I}^{\mathsf{aux}}$, there must be an assertion $S(\tau(t), \tau(t')) \in \mathcal{A}_{Q_\mathsf{R}} \cup \mathcal{A}_{Q_{\iota(j)}}$ such that $\mathcal{O} \models S \sqsubseteq R$; $\tau(t) = e$ if $\pi(t) = e^j$. By the definition of $\mathcal{A}_{Q_\mathsf{R}}$, which is based on consequences (see Definition 6.2), we get $R(\tau(t), \tau(t')) \in \mathcal{A}_{Q_\mathsf{R}}$. Hence, $\mathcal{I}_i \models \mathcal{A}_{Q_\mathsf{R}}$ implies $(\pi'(t), \pi'(t')) \in R^{\mathcal{I}_i}$.

- R is rigid and one of $\pi(t)$ and $\pi(t')$ is contained in $\Delta_{\mathsf{t}}^{\mathcal{I}_j}$:
  We argument similar to the previous case. Again, $(\pi(t), \pi(t')) \in R^{\mathcal{J}_i}$ implies $(\pi(t), \pi(t')) \in R^{\mathcal{I}_j}$. By Definition 2.11, since only $\mathcal{A}_{R_\mathsf{F}}$ contains assertions on elements from $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$, and because $\mathcal{A}_{R_\mathsf{F}}$ contains all rigid assertions on the elements of $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ that follow by $\mathcal{O}$ (see the definition of $\mathcal{A}_{R_\mathsf{F}}$), there must be an assertion $R(\tau(t), \tau(t')) \in \mathcal{A}_{R_\mathsf{F}}$. From $\mathcal{I}_i \models \mathcal{A}_{R_\mathsf{F}}$, we then get $(\pi'(t), \pi'(t')) \in R^{\mathcal{I}_i}$.

- R is flexible:
  Given $(\pi(t), \pi(t')) \in R^{\mathcal{J}_i}$ and the fact that witnesses are not defined for elements of Δ_n, there must be a rigid role S such that $(\pi(t), \pi(t')) \in S^{\mathcal{I}_j}$ and $\mathcal{O} \models S \sqsubseteq R$ by the definition of $\mathcal{J}_i$. As in the respective previous case (i.e., based on the kind of $\pi'(t)$ and $\pi'(t')$ here), it follows that $(\pi'(t), \pi'(t')) \in S^{\mathcal{I}_i}$. Since $\mathcal{I}_i \models \mathcal{O}$, we then obtain $(\pi'(t), \pi'(t')) \in R^{\mathcal{I}_i}$.

It remains to define π' for the variables of φ that are mapped by π into $\bigcup_{j=0}^{n+k} \Delta_{\mathsf{u}}^{\mathcal{I}_j}$. Since the relations in $\mathcal{J}_i$ are based on those in the canonical interpretations, Definition 2.11 implies that all variables that occur in role atoms together with a term mapped to an element $e \in \Delta_\mathsf{n}$ by π are of the form $u^j_{eS_0}$, and the role atom must be an R-atom such that $\mathcal{O} \models S_0 \sqsubseteq R$. Moreover, the assumption that φ is connected yields that, if any variable is mapped to an element $u^j_{eS_0 \dots S_\ell} \in \Delta_{\mathsf{u}}^{\mathcal{I}_j}$, then there is a variable that is mapped to $u^j_{eS_0}$, one directly connected to it and mapped to $u^j_{eS_0S_1}$, etc. We hence can proceed as follows.

2) We consider all $y \in \mathsf{N_V}(\varphi)$ for which there is an atom $R(t,y) \in \varphi$ where $\pi(t) \in \Delta_{\mathsf{n}}$ and $\pi(y) \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$, and assume $\pi(t) = e^j$, $\pi(y) = u^j_{eS_0}$, $S_0 \in \mathsf{N}^-_{\mathsf{R}}$, and $\mathcal{O} \models S_0 \sqsubseteq R$. Recall that $\pi'(t) = e^i$. The goal is to choose an element of $\Delta^{\mathcal{I}_i}$ as value for $\pi'(y)$ so that π' can be (extended to) a homomorphism of φ into $\mathcal{I}_i$. For now, we however only show that our definition of $\pi'(y)$ satisfies $(\pi'(t), \pi'(y)) \in R^{\mathcal{J}_i}$ regarding all these role atoms. The remaining atoms that contain y are then covered in Part 3).

If $i = j$, then we can directly define $\pi'(y) := \pi(y)$ by Lemmas 6.11 and 6.13. Otherwise, we distinguish the following two cases. Note that $(e^j, u^j_{eS_0}) \in R^{\mathcal{J}_i}$ implies $(e^j, u^j_{eS_0}) \in R^{\mathcal{I}_j}$ by Lemma 6.12.

- If $\mathcal{W}_{\mathcal{O}}(u^j_{eS_0}) \neq \emptyset$, then $(e^j, u^j_{eS_0}) \in R^{\mathcal{I}_j}$ implies $(e^j, u^j_{eS_0}) \in R^{\mathcal{J}_i}$ by the definition of $\mathcal{J}_i$. We then get $e^i = \pi'(t) \in (\exists S_0)^{\mathcal{I}_i}$ by Corollary 6.16. According to Definition 2.11, the element $u^i_{eS_0}$ then exists and the pair $(e^i, u^i_{eS_0})$ is related in $\mathcal{I}_i$ as $(e^j, u^j_{eS_0})$ is related in $\mathcal{I}_j$. And the latter tuple is interpreted in the same way in $\mathcal{J}_i$. We can thus set $\pi'(y) := u^i_{eS_0}$.

- If $\mathcal{W}_{\mathcal{O}}(u^j_{eS_0}) = \emptyset$, then the definition of $\mathcal{J}_i$ yields that, for all atoms $R(t,y)$ as above, there is a rigid role S such that $\mathcal{O} \models S_0 \sqsubseteq S$, $\mathcal{O} \models S \sqsubseteq R$, and $(e^j, u^j_{eS_0}) \in S^{\mathcal{J}_i}$. Note that S_0 must be flexible since otherwise $\exists S_0$ would be a witness for $u^j_{eS_0}$. Furthermore, since $(e^j, u^j_{eS_0}) \in S_0^{\mathcal{I}_j}$, Lemma 2.14 yields that the assertion $\exists S_0(e)$ is a consequence of the basic concepts obtained from the assertions involving e in $\mathcal{K}^j_{\mathsf{R}}$. We now show by a case distinction on the kind of e that this is still the case if $\mathcal{A}_{R_{\mathsf{F}}}$ is disregarded.
  - If $e \in \mathsf{N_I}(\mathcal{K})$, then we have that any (rigid) basic concept assertion on e that is a consequence of $\mathcal{A}_{R_{\mathsf{F}}}$, by taking $\mathcal{O}$ into account, must be contained in $\mathcal{A}_{\mathsf{R}}$, since $\mathcal{A}_{\mathsf{R}}$ is an ABox type and $\mathcal{I}_j$ is a model of both these ABoxes. Since $\mathcal{A}_{R_{\mathsf{F}}}$ does not contain flexible assertions, $\exists S_0(e)$ is also a consequence of $\mathcal{K}^j_{\mathsf{R}}$ if $\mathcal{A}_{R_{\mathsf{F}}}$ is disregarded.
  - If $e \in \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$ is of the form $e = a_x$ and φ' is the CQ in which x appears, then we know that $\exists S_0(e)$ is a consequence of the assertions in $\mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(j)}} \cup \mathcal{A}_{R_{\mathsf{F}}}$ (i.e., again, taking $\mathcal{O}$ into account). Regarding $\mathcal{A}_{R_{\mathsf{F}}}$, we thus consider a rigid concept assertions $\exists R'(a_x)$, $R' \in \mathsf{N}^-_{\mathsf{R}}$, for which there is a flexible assertion $\exists S'(a_x) \in R_{\mathsf{F}}$ such that $\mathcal{O} \models S' \sqsubseteq \exists R'$. By Condition (C6), all those assertions $\exists R'(a_x)$ follow however from some set of assertions $\mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(j')}}$, $j' \in [0, n+k]$, and hence from $\bigsqcap \mathbb{B}(\varphi', a_x)$. By the definition of $\mathcal{A}_{Q_{\mathsf{R}}}$, they are thus contained in this ABox, which means that $\mathcal{A}_{R_{\mathsf{F}}}$ can be disregarded.
  - If $e \in \mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$, then $\exists S_0(e)$ must similarly follow from ABox assertions by Lemma 2.14; particularly, it follows exclusively from $\mathcal{A}_{R_{\mathsf{F}}}$ (and $\mathcal{O}$) because elements from $\mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$ do not occur in other ABoxes. Since $\mathcal{A}_{R_{\mathsf{F}}}$ contains only rigid assertions, the corresponding rigid basic concepts constitute a witness for $u^j_{eS_0}$, which contradicts our assumption and yields $e \notin \mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$.

  The case distinction proves the entailment required to apply the "only if" direction Condition (C6) to infer that $\exists S_0(e) \in \mathcal{A}_{R_{\mathsf{F}}}$. Since $\mathcal{I}_i \models \mathcal{A}_{R_{\mathsf{F}}}$, $(e^i, a^i_{eS_0}) \in S^{\mathcal{I}_i}$ holds for all rigid roles S as above. Given $\mathcal{I}_i \models \mathcal{O}$, the above assumptions on $\mathcal{O}$, and

$\mathcal{W}_\mathcal{O}(u^j_{eS_0}) = \emptyset$, $(e^i, a^i_{eS_0})$ satisfies all the role atoms $R(t, y)$ in $\mathcal{I}_i$ that are mapped to $(e^j, u^j_{eS_0})$ by π. We can therefore define $\pi'(y) := a^i_{eS_0}$.

It thus remains to consider the satisfaction of the atoms we left out in 2) and, in particular, the other variables of φ mapped by π to elements of $\bigcup_{j=0}^{n+k} \Delta^{\mathcal{I}_j}_{\mathsf{u}}$. As described above, we can assume them to be related in a tree structure and, especially, to the elements we focused on in 2).

3) We finish the definition of π' using an induction over the structures of unnamed elements in the image of the homomorphism π, starting with the elements we considered in 2). For all variables y with $\pi(y) \in \Delta^{\mathcal{I}_i}_{\mathsf{u}}$, we can obviously set $\pi'(y) := \pi(y)$. We therefore only consider the case that $\pi(y) \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ and $j \neq i$ in the following. Note that this is valid for the induction by Fact 6.10 and the interpretation of roles in $\mathcal{J}_i$, which show that elements from different sets $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$ and $\Delta^{\mathcal{I}_j}_{\mathsf{u}}$ cannot be related in $\mathcal{J}_i$.

Given the latter observations, we can also maintain the following invariant while finishing the construction of π' regarding the remaining variables y (i.e., we do not have to satisfy the invariant at all for variables mapped by π to elements from $\Delta^{\mathcal{I}_i}_{\mathsf{u}}$ since π' is already defined for all variables directly connected to them); let m denote the number of variables for which π' is defined already at the respective moments in the induction: If $\pi(y) = u^j_{\varrho S_1}$, then either (i) $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) = \emptyset$, $\pi'(y) = a^i_{\varrho S_1}$, and $|\varrho| < m$, or (ii) $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) \neq \emptyset$, $\pi'(y)$ is of the form $u^i_{\sigma S_1}$, and $|\sigma| < m$. As induction hypotheses, we assume that the (partial) definition of π' satisfies all role atoms that only contain variables for which it is already defined and the invariant. It can readily be checked that our definitions from 2), which represent the base case, satisfy both these requirements.

To show that π' is a homomorphism of φ into $\mathcal{I}_i$ regarding the variables mapped so far, it remains to consider the concept atoms. We assume $\pi(y) = u^j_{\varrho S_1} \in \Delta^{\mathcal{I}_j}_{\mathsf{u}}$ and $\pi'(y)$ to be defined already and consider all concept atoms $A(y) \in \varphi$. Since $u^j_{\varrho S_1} \in A^{\mathcal{J}_i}$, we know by Lemma 6.13 that either (i') there is a $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$ with $u^j_{\varrho S_1} \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ and $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq A$, or (ii') $u^j_{\varrho S_1} \in A^{\mathcal{I}_j}$ and $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) \neq \emptyset$. If the above Case (ii) applies, meaning $\pi'(y) = u^i_{\sigma S_1}$, then two applications of Lemma 2.15 yield that $\mathcal{O} \models \exists S_1^- \sqsubseteq A$ and $\pi'(y) = u^i_{\sigma S_1} \in A^{\mathcal{I}_i}$. Otherwise, (i') must hold because of $u^j_{\varrho S_1} \in A^{\mathcal{J}_i}$ by the definition of $\mathcal{J}_i$. $\mathcal{I}_i \models \mathcal{O}$ then implies $u^j_{\varrho S_1} \in A^{\mathcal{I}_j}$, and Lemma 2.15 yields $\mathcal{O} \models \exists S_1^- \sqsubseteq \bigsqcap \mathcal{B}$. From $\pi'(y) = a^i_{\varrho S_1}$, we then get $a^i_{\varrho S_1} \in (\bigsqcap \mathcal{B})^{\mathcal{I}_i}$ by the definition of $\mathcal{A}_{R_\mathsf{F}}$. Given $\mathcal{I}_i \models \mathcal{O}$, we conclude that $\pi'(y) \in A^{\mathcal{I}_i}$.

To continue the definition of π', we consider an element $u^j_{\varrho S_1 S_2} \in \mathsf{range}(\pi)$ and all role atoms $R(y, z) \in \varphi$ with $\pi(y) = u^j_{\varrho S_1}$ and $\pi(z) = u^j_{\varrho S_1 S_2}$. We hence can assume that φ contains a variable x such that $\pi(x) = u^j_{\varrho S_1}$ for which π' has been defined already. For all these role atoms, we have that $\mathcal{O} \models S_2 \sqsubseteq R$ by the definition of $\mathcal{J}_i$ and Definition 2.11. Once again, we distinguish two cases w.r.t. $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1 S_2})$.

- If $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1 S_2}) \neq \emptyset$, then we have $(u^j_{\varrho S_1}, u^j_{\varrho S_1 S_2}) \in S_2^{\mathcal{J}_i}$ since the pair is contained in $S_2^{\mathcal{I}_j}$ and also $\mathcal{O} \models \exists S_1^- \sqsubseteq \exists S_2$ by the definitions of the two interpretations. Further, one of the following two cases must apply:
  - If $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) \neq \emptyset$, then (ii) implies that $\pi'(y)$ is of the form $\pi'(y) = u^i_{\sigma S_1}$. Given $\mathcal{I}_i \models \mathcal{O}$, the element $u^i_{\sigma S_1 S_2}$ then exists, and $(u^i_{\sigma S_1}, u^i_{\sigma S_1 S_2})$ satisfies all role atoms $R(y, z)$ of the above form in $\mathcal{I}_i$ by Definition 2.11. Hence, we

can define $\pi'(z) := u^i_{\sigma S_1 S_2}$ for all such variables z and maintain the invariant (Case (ii)).

– If $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) = \emptyset$, then there must be a set $\mathcal{B} \subseteq \mathbb{B}_\mathsf{R}(\mathcal{O})$ such that $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists S_2$ and $u^j_{\varrho S_1} \in (\bigsqcap \mathcal{B})^{\mathcal{I}_j}$ by Definition 6.4, which implies $\mathcal{O} \models \exists S_1^- \sqsubseteq \bigsqcap \mathcal{B}$ by Lemma 2.15. Since (i) yields that $\pi'(y)$ is of the form $a^i_{\varrho S_1}$, the definition of $\mathcal{A}_{R_\mathsf{F}}$ implies $\mathcal{B}(a_{\varrho S_1}) \subseteq \mathcal{A}_{R_\mathsf{F}}$, and $\mathcal{I}_i \models \mathcal{A}_{R_\mathsf{F}}$ then yields $\pi'(y) \in (\bigsqcap \mathcal{B})^{\mathcal{I}_i}$. Together with $\mathcal{I}_i \models \mathcal{O}$, this implies $\pi'(y) \in (\exists S_2)^{\mathcal{I}_i}$. By Definition 2.11, the element $u^i_{a_{\varrho S_1} S_2}$ thus exists and the pair $(a^i_{\varrho S_1}, u^i_{a_{\varrho S_1} S_2})$ satisfies all the roles R in $\mathcal{I}_i$. We hence can set $\pi'(z) := u^i_{a_{\varrho S_1} S_2}$; this satisfies Case (ii) of our invariant.

- If $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1 S_2}) = \emptyset$, then we know that also $\mathcal{W}_\mathcal{O}(u^j_{\varrho S_1}) = \emptyset$ holds, and hence Case (i) of our invariant applies, meaning $\pi'(y) = a^i_{\varrho S_1}$. In addition, for every of the above role atoms $R(y, z)$, there is a rigid role S such that $\mathcal{O} \models S \sqsubseteq R$ and $(u^j_{\varrho S_1}, u^j_{\varrho S_1 S_2}) \in S^{\mathcal{I}_j}$ by the definition of $\mathcal{J}_i$. Definition 2.11 then yields $\mathcal{O} \models S_2 \sqsubseteq S$. Together with $\mathcal{O} \models \exists S_1^- \sqsubseteq \exists S_2$ and the given bound on the length of ϱ, this implies that the element $a^i_{\varrho S_1 S_2}$ exists in $\mathcal{A}_{R_\mathsf{F}}$, in all the rigid assertions $S(a^i_{\varrho S_1}, a^i_{\varrho S_1 S_2})$. Thus, $\mathcal{I}_i \models \mathcal{A}_{R_\mathsf{F}}$ and $\mathcal{I}_i \models \mathcal{O}$, because of the fact that $\mathcal{O} \models S \sqsubseteq R$ yield that $(a_{\varrho S_1}, a_{\varrho S_1 S_2})$ satisfies all all relevant role atoms $R(y, z)$. We set $\pi'(z) := a_{\varrho S_1 S_2}$ for all such variables z and obtain Case (i) of our invariant.

This concludes the construction of π' and shows that it is a homomorphism of φ into $\mathcal{I}_i$, which contradicts our assumption that $\mathcal{I}_i \models \neg\varphi$. □

This finishes also the proof of Lemma 6.9. In what follows, we use this characterization of r-satisfiability for obtaining complexity bounds for both combined and data complexity.

6.2 Combined Complexity

In this section, we show that TCQ entailment is in PSpace w.r.t. combined complexity, which matches the hardness given by satisfiability in LTL. The previous section shows that there is no need to store the exponentially large set $\mathcal{W}$ for testing the conditions characterizing r-satisfiability. Hence, we can apply Algorithm 3.1: we first guess a tuple $(\mathcal{A}_\mathsf{R}, Q_\mathsf{R}, Q_\mathsf{R}^\neg, R_\mathsf{F})$ as described in Section 6.1 and then test the satisfiability of the LTL formula Φ^{pa} as it is done in Algorithm 2.1, but ensure additionally that the guessed worlds satisfy the conditions of r-completeness (see Definition 6.8). This procedure is in line with Lemma 3.13: we integrate the r-satisfiability with the t-satisfiability test.

Subsequently, we specify the algorithm, prove correctness, and then show that it uses only polynomial space. Note that, throughout this section, all complexity considerations target combined complexity so that we do not always mention that explicitly.

We start describing the functions we apply.

- `CQCONSEQUENCES`: Given a CQ φ and an ontology $\mathcal{O}$, it computes the set $\mathcal{C}_\mathcal{O}(\varphi)$ (see Definition 6.2). That is, it performs a series of subsumption tests.

Since the size of the considered sets $\mathbb{B}^-_{\mathsf{R}}(\mathcal{O})$ and $\mathsf{N}^-_{\mathsf{RR}}(\mathcal{O})$ depends polynomially on the size of $\mathcal{O}$; that of $\mathcal{A}_\varphi$, also regarded, depends linearly on the size of φ; and subsumption in *DL-Lite*$^{\mathcal{H}}_{horn}$ is in P w.r.t. combined complexity [Art+09, Thm. 8.2], this procedure runs in polynomial time in the size of its input.

- `KBCONSISTENT`: Given a knowledge base in *DL-Lite*$^{\mathcal{H}}_{horn}$, it decides consistency.

  According to [Art+09, Thm. 8.2], this can be done in polynomial time in the size of the input.

- `UCQENTAILED`: Given a CQ φ and a *DL-Lite*$^{\mathcal{H}}_{horn}$ knowledge base $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$, it decides the entailment $\mathcal{K} \models \varphi$ based on the nondeterministic version of the algorithm in [BAC10] (see the sketch after Theorem 12 in that paper) in three steps:
  - apply algorithm PerfectRef [BAC10, Fig. 2] to rewrite φ using $\mathcal{O}$,
  - nondeterministically choose a CQ ψ from the resulting UCQ,
  - check whether $\mathrm{DB}(\mathcal{A}) \models \psi$.

  Especially note that the size of ψ is polynomial in that of φ. Altogether, this nondeterministic procedure runs in polynomial time in its input [BAC10].

Next to the above functions, we use two enumerators:

- `TREEWITNESSENUM`: Given a CQ φ and an ontology $\mathcal{O}$, it enumerates all tree witnesses for φ w.r.t. $\mathcal{O}$ (see Definition 6.3).

  Since tree witnesses mimic the structure of φ and the range of the candidate mappings $\mathrm{f}\colon \mathsf{N}_{\mathsf{V}}(\varphi) \to (\mathsf{N}^-_{\mathsf{R}} \times 2^{\mathsf{N}^-_{\mathsf{R}}})^*$ contains only sequences of size polynomial in the size of φ, all these candidates can be enumerated using polynomial space only. The conditions of Definition 6.3 can also be checked in polynomial time; regarding the subsumption, we again refer to [Art+09, Thm. 8.2].

- `WITNESSQENUM`: Given a CQ φ, an ontology $\mathcal{O}$, and a tree witnesses f for φ w.r.t. $\mathcal{O}$, it enumerates all witness queries for φ w.r.t. $\mathcal{O}$ and f (see Definition 6.5).

  All sets $\mathcal{B} \subseteq \mathbb{B}(\mathcal{O})$ are of polynomial size and can hence be enumerated using only polynomial space, in the size of $\mathcal{O}$. Checking if $\mathcal{B} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ (see Definition 6.3) is again subsumption testing and hence in P.

  (i) Tree witness queries can, based on φ, f, and $\mathcal{B}$, be constructed by considering the polynomially-sized f, enumerating all possibilities of assigning a set of basic concepts to each relevant node of φ, and checking if the conditions from Definition 6.5 are satisfied. The size of the set is polynomial in $\mathcal{O}$, and the testing again involves a series of P-tests.

  (ii) The other kind of witness query can be constructed as follows:[8]
  - Construct a graph over all roles in $\mathsf{N}^-_{\mathsf{R}}(\mathcal{O})$ that contains an edge from R to S iff $\mathcal{O} \models \exists R^- \sqsubseteq \exists S$.

[8] This idea is also implicitly used in the form of the reachability relation $\rightsquigarrow$ in [BBL05; KKS12].

– Check whether it is possible to reach an S satisfying $\{\exists S\} \in \mathrm{Con}_{\mathcal{O}}(\varphi, \mathrm{f})$ from an R with $\mathcal{O} \models \bigsqcap \mathcal{B} \sqsubseteq \exists R$ for some $\mathcal{B} \subseteq \mathbb{B}_{\mathsf{R}}(\mathcal{O})$.

It is easy to see that this approach is both sound and complete.

The graph can be constructed by quadratically many P-tests. The subsequent reachability test can, given the above observations, be done within polynomial space. More specifically, this is because $\mathcal{B}$, R, and S can be guessed and then considered by using polynomial space only and the reachability problem is in NLogSpace (e.g., see [AB09, p. 74]).

The above observations show that all the procedures we apply run in polynomial space in the size of their input. Our algorithm based on Algorithm 2.1 is presented next.

Definition 6.17 Given a TCQ Φ and TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$, satisfiability of Φ w.r.t. $\mathcal{K}$ can be decided by running Algorithm 3.1:

- `GUESSDATA`: It guesses a tuple $(\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q^{\neg}_{\mathsf{R}}, R_{\mathsf{F}})$ with an ABox type $\mathcal{A}_{\mathsf{R}}$ for $\mathcal{K}$ and sets $Q_{\mathsf{R}}, Q^{\neg}_{\mathsf{R}} \subseteq \mathcal{Q}_{\Phi}$ and $R_{\mathsf{F}} \subseteq \{\exists S(b) \mid S \in \mathsf{N}_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}_{\mathsf{RR}},\ b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}\}$;

  Additionally, it initializes a set $R^{\Leftarrow}_{\mathsf{F}} := R_{\mathsf{F}}$ and returns it together with the guessed tuple.

- `TESTRSAT`: Given $\Phi, \mathcal{O}, \mathcal{A}_i, d$ (i.e., the tuple guessed and $R^{\Leftarrow}_{\mathsf{F}}$), i, s, p, and W, it first defines the following sets:

$$\begin{aligned} Q_W &:= \{\varphi_j \in \mathcal{Q}_{\Phi} \mid p_j \in W\}, \\ \mathcal{A}_{Q_{\mathsf{R}}} &:= \bigcup_{\varphi \in Q_{\mathsf{R}}} \texttt{CQCONSEQUENCES}(\varphi, \mathcal{O}), \\ \mathcal{K}_{\mathsf{R}} &:= \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_i \rangle, \end{aligned}$$

  and, for every assertion $\exists S(b) \in R^{\Leftarrow}_{\mathsf{F}}$, updates $R^{\Leftarrow}_{\mathsf{F}} := R^{\Leftarrow}_{\mathsf{F}} \setminus \{\exists S(b)\}$ if the function $\texttt{UCQENTAILED}(\exists S(b), \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_i \rangle)$ returns *true*.

  Then, it returns *true* iff the following conditions are satisfied:

  (C1) Check if $\texttt{KBCONSISTENT}(\mathcal{K}_{\mathsf{R}})$ returns *true*.

  (C2) For each $p_j \in \overline{W}$:
  - Guess a connected set $\mathcal{A}'_{R_{\mathsf{F}}} \subseteq \mathcal{A}_{R_{\mathsf{F}}}$ that is of polynomial size (the size of φ_j).
  - Check if $\texttt{UCQENTAILED}(\varphi_j, \langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}'_{R_{\mathsf{F}}} \cup \mathcal{A}_i \rangle)$ returns *false*.

  (C3) For each $p_j \in W$: Check if $\varphi_j \in Q_{\mathsf{R}}$.

  (C4) For each $p_j \in \overline{W}$: Check if $\varphi_j \in Q^{\neg}_{\mathsf{R}}$.

  (C5) For all $\varphi \in Q^{\neg}_{\mathsf{R}}$, f in $\texttt{TREEWITNESSENUM}(\varphi, \mathcal{O})$, and ψ in $\texttt{WITNESSQENUM}(\varphi, \mathrm{f}, \mathcal{O})$:
  Check if $\texttt{UCQENTAILED}(\mathcal{K}_{\mathsf{R}}, \psi)$ returns *false*.

  (C6) For each $S \in \mathsf{N}_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}_{\mathsf{RR}}$ and $b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \cup \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$:
  If $\texttt{UCQENTAILED}(\exists S(b), \mathcal{K}_{\mathsf{R}})$ returns *true*, then check if $\exists S(b) \in R_{\mathsf{F}}$.
  If $i = s + p$, then check if $R^{\Leftarrow}_{\mathsf{F}} = \emptyset$. ♢

Observe that our algorithm modifies Algorithm 2.1 in two critical points: it adapts the recording of the start of the period, and it adds tests that may lead to a negative outcome. Most of the tests we apply are self-explanatory. Note however that the set $R_{\mathsf{F}}^{\Leftarrow}$ is used to check the "only if" direction of Condition (C6), which cannot be done locally (i.e., separately for each time point i). The next lemma summarizes the goal of our extensions.

Lemma 6.18 *The nondeterministic algorithm described in Definition 6.17 decides TCQ satisfiability w.r.t. a TKB by using only polynomial space w.r.t. combined complexity.*

Proof. For proving correctness, we consider the conditions in Lemma 3.13.

- Let the set $\mathcal{W} = \{W_1, \dots, W_k\}$ be defined as the set of all worlds W encountered during a run of the procedure. The mapping $\iota\colon [0, n] \to [1, k]$ is defined such that $\iota(i) := \ell$ where $W_\ell = \mathcal{F}_c \cap \{p_1, \dots, p_m\}$.

- Regarding t-satisfiability (see Definition 3.11), it is easy to see that the above definition of $\mathcal{W}$ fulfills the first condition. For the second condition, we can focus on our modifications since Algorithm 2.1 is correct by Lemma 2.21.

  $s > n$: The (possible) moving of the recorded period start may cause Algorithm 2.1 to run longer, though maximally n iterations of that algorithm, but does not change its final outcome.

  `GUESSDATA/TESTRSAT`: Our other extensions do not modify steps in Algorithm 2.1 itself nor do they ever return true on their own (i.e., independent of Algorithm 2.1), hence our algorithm is sound w.r.t. the second condition as well; and it is complete if the extensions never return false if an r-satisfiable set $\mathcal{W}$ exists.

  The last condition for t-satisfiability is satisfied also because of our definitions of ι and $\mathcal{W}$.

- It thus remains to show that $\mathcal{W}$ is r-satisfiable iff the extensions do *not* return false on their own. By Lemma 6.9, we can consider Conditions (C1)–(C6) from Definition 6.8.

  (C1) Observe that our algorithm only checks the consistency of the knowledge base $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_i \rangle$ and hence drops $\mathcal{A}_{R_{\mathsf{F}}}$.

  However, this exponentially large ABox can be ignored for this consistency test since, once Condition (C6) is verified, we know that for each $\mathcal{A}_{\exists S(b)} \subseteq \mathcal{A}_{R_{\mathsf{F}}}$ there is at least one index $j \in [0, n + k]$ for which the existence of the elements described in $\mathcal{A}_{\exists S(b)}$ follows from the KB $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{W_{\iota(j)}}} \cup \mathcal{A}_j \rangle$ where, for all $j > n$, $W_{\iota(j)} = W_j$ and $\mathcal{A}_j = \emptyset$. Hence, the rigid consequences of the assertion $\exists S(b)$ regarding $b \in \mathsf{N_I}(\mathcal{K}) \cup \mathsf{N_I^{aux}}$, which are represented by $\mathcal{A}_{\exists S(b)}$, must follow from $\mathcal{A}_{\mathsf{R}}$ or $\mathcal{A}_{Q_{\mathsf{R}}}$ (i.e., based on the kind of b) given our definitions of all the additional ABoxes. We thus can disregard the assertions from $\mathcal{A}_{\exists S(b)}$ including the elements from $\mathsf{N_I^{tree}}$ since any model of the KB we consider must have domain elements of this kind.

  Further note that we modify the initialization of the period start s, in order to ensure that all the ABoxes in the given TKB are considered; but the result

of [SC85, Thm. 4.7] only applies when there are no external conditions on the propositional models. It is however easy to see that the latter result extends to our case.

(C2) To check whether $\mathcal{K}_{\mathsf{R}} \not\models \varphi_j$ holds, for each $p_j \in \overline{W}$, we apply the algorithm `UCQENTAILED`. Given the fact that this nondeterministic algorithm basically checks the satisfaction of a CQ whose size is polynomial in that of φ_j, we can especially consider the exponentially large, forest-shaped ABox $\mathcal{A}_{R_{\mathsf{F}}}$ by regarding only a nondeterministically chosen part of size polynomial in φ_j.

(C3) The corresponding test obviously captures this condition.

(C4) The corresponding test obviously captures this condition.

(C5) Given the above specifications of the subprocedures used, our tests also capture this condition.

(C6) The "if" direction of the equivalence is captured by the first test. The other direction of Condition (C6) is checked globally, regarding all time points considered. Observe that the latter test may require us to look for an LTL structure with a longer period. Though, the maximally required period is still exponential in the input—in contrast to the original dependency of the period length on the given formula, the TKB is now also of influence—and can be represented in polynomial space. To see this, note that our global condition corresponds to the extension of the given TCQ with linearly many additional conjuncts of the form $\Diamond_P \Diamond_F \exists S(b)$: if there is a model of Φ w.r.t. $\mathcal{K}$, then a set R_{F} so that the algorithm does not fail because of that adaptation of the query can easily be defined; completeness is retained since, if the algorithm succeeds, then we can construct such a model for that TCQ based on its processing.

We analyze the complexity. The nondeterministic guessing of the polynomially large sets $\mathcal{A}_{\mathsf{R}}, Q_{\mathsf{R}}, Q_{\mathsf{R}}^{\neg}$, and R_{F} can be done using polynomial space only.

The number of sets $\mathcal{C}_{\mathcal{O}}(\varphi_j)$ to be computed for the construction of $\mathcal{A}_{Q_{\mathsf{R}}}$ equals the cardinality of Q_{R}, which depends linearly on the size of Φ. According to [Art+09, Thm. 8.2], we need a number of P (subsumption) tests to compute each set $\mathcal{C}_{\mathcal{O}}(\varphi_j)$; and this number depends linearly on the size of φ_j (and hence that of Φ) and polynomially on the size of the ontology $\mathcal{O}$.

Regarding the original parts (of Algorithm 2.1), we refer to Lemma 2.21.

It remains to consider the tests in the last extension. The above descriptions of the applied subprocedures show that all of them run within polynomial space w.r.t. combined complexity. Since we apply them only for input of size polynomial in Φ and $\mathcal{K}$, the guessing for testing Condition (C2) can clearly be done nondeterministically in polynomial time, and all other checks described for testing the conditions can be done in polynomial time, our nondeterministic algorithm altogether uses only polynomial space. □

Given the general consideration of rigid names, we thus can conclude this section with a result which is even better than that for $\mathcal{EL}$ (see Corollary 5.16). The nondeterminism is

not relevant for PSPACE complexity according to the well-known result of Savitch [Sav70, Thm. 1].

Corollary 6.19 *TCQ entailment in DL-Lite$^{\mathcal{H}}_{horn}$ is in* PSPACE *in combined complexity, even if* $\mathsf{N}_{\mathsf{RR}} \neq \emptyset$.

6.3 First-Order Rewritings of r-Satisfiability

In this section, we show that r-satisfiability is first-order rewritable and thus, once again, solve the following problem: Given a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ with $\mathcal{W} = \{W_1, \dots, W_k\}$, a mapping $\iota\colon [0,n] \to [1,k]$, and the TKB $\mathcal{K} = \langle \mathcal{O}, \mathfrak{A} \rangle$ in *DL-Lite*$^{\mathcal{H}}_{horn}$ with $\mathfrak{A} = (\mathcal{A}_i)_{0 \leq i \leq n}$, is $\mathcal{W}$ r-satisfiable w.r.t. ι and $\mathcal{K}$? More precisely:

- We propose a set of FO formulas such that, if evaluated over the ABox sequence considered as one finite structure TDB($\mathfrak{A}$), *all* of them are satisfied iff $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$.
- The formulas capture the conditions for r-completeness (see Definition 6.8)—our approach is based on Lemma 6.9.
- In particular, they address a fixed tuple as described in Section 6.1.2, which we construct based on $\mathcal{W}$.

The idea is thus to check if the this tuple is r-complete for deciding the r-satisfiability of $\mathcal{W}$. The correctness of this approach is established at the end of this section (see Lemma 6.30). Observe that the tests for r-completeness basically represent a series of consistency and non-entailment tests regarding traditional KBs. We can hence apply the two rewritings proposed by [BAC10], which are Boolean UCQs (with inequalities)—called $q_{unsat(\mathcal{O})}$ and $\mathsf{PerfectRef}(\varphi, \mathcal{O})$, respectively—, independent of the given ABox $\mathcal{A}$, and evaluated over the FO structure DB($\mathcal{A}$) to solve these problems (see Lemma 2.24). However, the conditions for r-completeness target many more ABoxes in addition to the sequence $\mathfrak{A}$. This section therefore focuses on the incorporation of this data into the two existing rewritings, so that the resulting formulas can be answered over $\mathfrak{A}$ alone. Then, the r-completeness conditions can be formulated easily (see Lemma 6.30). To get an intuition of our goal and proceeding, consider the following example.

Example 6.20 We focus on the atemporal case and the rewriting $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ of UCQ entailment. Recall that, for a KB $\langle \mathcal{O}, \mathcal{A}_i \rangle$ and CQ φ, Lemma 2.24 yields the following:

$$\langle \mathcal{O}, \mathcal{A}_i \rangle \models \varphi \text{ iff } \mathrm{DB}(\mathcal{A}_i) \models \mathsf{PerfectRef}(\varphi, \mathcal{O}).$$

Let now $\mathcal{A} := \{A(a)\}$ be an additional ABox that is relevant to test r-completeness but not part of the given KB; for simplicity, we assume a to also occur in $\mathcal{A}_i$.[9] The goal is then to adapt $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ such that the resulting FO formula $\mathsf{PRef}_{(\varphi, \mathcal{O}|\mathcal{A})}$ is as follows:

$$\langle \mathcal{O}, \mathcal{A}_i \cup \mathcal{A} \rangle \models \varphi \text{ iff } \mathrm{DB}(\mathcal{A}_i) \models \mathsf{PRef}_{(\varphi, \mathcal{O}|\mathcal{A})}.$$

[9] We detail later in this section why this assumption considerably simplifies matters. The point is that the quantifiers in the queries we adapt quantify only over the individuals in the given KB. In order to refer to new individuals, the quantification thus has to be adapted as well.

This can be done, for instance, by replacing every occurrence of $A(a)$ in $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ by *true*, and every occurrence of an atom $A(x)$ with variable x by $(x = a) \vee A(x)$. ♢

The fact that we consider different kinds of auxiliary ABoxes focusing on various kinds of named individuals makes the actual rewritings yet more complex than the example case. Moreover, these rewritings are evaluated over the structure $\mathrm{TDB}(\mathfrak{A})$. This temporal database interpretation is specified analogously to $\mathrm{DB}(\mathcal{A})$, the structure considered in the atemporal case, which regards only a single ABox $\mathcal{A}$.

Definition 6.21 (TDB($\mathfrak{A}$)) For the given ABox sequence $\mathfrak{A} = (\mathcal{A}_i)_{0 \leq i \leq n}$, the two-sorted first-order structure $\mathrm{TDB}(\mathfrak{A}) = (\mathsf{N_I}(\mathfrak{A}), \mathsf{N_T}(\mathfrak{A}), \cdot^{\mathrm{TDB}})$ over the *individual domain* $\mathsf{N_I}(\mathfrak{A}) := \bigcup_{0 \leq i \leq n} \mathsf{N_I}(\mathcal{A}_i)$ and *temporal domain* $\mathsf{N_T}(\mathfrak{A}) := [-1, n]$ contains the following relations for all $B \in \mathbb{B}(\mathfrak{A})$ and $R \in \mathsf{N_R}(\mathfrak{A})$:[10]

$$\mathsf{B}^{\mathrm{TDB}} := \{(a, i) \mid i \in [0, n],\ B(a) \in \mathcal{A}_i\},$$
$$\mathsf{R}^{\mathrm{TDB}} := \{(a, b, i) \mid i \in [0, n],\ R(a, b) \in \mathcal{A}_i\}.$$ ♢

The semantics of the satisfaction relation $\models$ is defined as usual:

$$\mathrm{TDB}(\mathfrak{A}) \models \mathsf{B}(a, i) \quad \text{iff} \quad (a, i) \in \mathsf{B}^{\mathrm{TDB}},$$
$$\mathrm{TDB}(\mathfrak{A}) \models \mathsf{R}(a, b, i) \quad \text{iff} \quad (a, b, i) \in \mathsf{R}^{\mathrm{TDB}}.$$

Note that we use the temporal domain element -1 to describe an empty ABox $\mathcal{A}_{-1} = \emptyset$, which can be accessed with formulas such as $\mathsf{B}(a, -1)$ and $\mathsf{R}(a, b, -1)$. Further, we can simplify presentation by omitting the sequence of ABoxes $\mathfrak{A}$ in the notation $\cdot^{\mathrm{TDB}}$, since it is fixed. Since $R(a, b)$ and $R^-(b, a)$ are used interchangeably for role atoms in the previous sections, we continue with this convention in the following, so that we may use atoms of the form $\mathsf{R}^-(b, a, i)$—especially in the rewriting—and assume the atoms to be resolved as intended when evaluated w.r.t. $\mathrm{TDB}(\mathfrak{A})$.

In Section 6.3.1, we next first specify the tuple we target and, based on that, present the KBs in focus of the r-completeness tests (i.e., for practical reasons, we have to adapt the KBs introduced in Definition 6.8). Then, we develop our versions of $q_{unsat(\mathcal{O})}$ and $\mathsf{PerfectRef}(\varphi, \mathcal{O})$, which address these KBs, in Section 6.3.2. Lastly, we present the final rewritings and prove the correctness of our approach in Section 6.3.3.

Note that we focus on data complexity throughout the section; if we refer to size etc., we hence disregard the impact of the ontology and the query. Constant size then indicates, for instance, that the sets are rather easily constructed or stored in a procedure deciding TCQ satisfiability (i.e., w.r.t. time and memory costs).

6.3.1 A Tuple for Testing r-Satisfiability

The first goal is to specify a fixed tuple $(\mathcal{A}_{\mathsf{R}[\mathcal{W}, \mathcal{B}_\Phi]}, Q_{\mathsf{R}[\mathcal{W}]}, Q^{\neg}_{\mathsf{R}[\mathcal{W}]}, R_{\mathsf{F}[\mathcal{W}, \mathcal{B}_\Phi]})$ as described in Section 6.1.2: it is based on two given sets $\mathcal{W} \subseteq 2^{\{p_1, \ldots, p_m\}}$ such that $\mathcal{W} = \{W_1, \ldots, W_k\}$ and $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N_I}(\Phi)\}$, both of constant size, and r-complete iff $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$. For testing r-completeness, we subsequently also specify the KBs regarded in these tests.

[10] Note that we follow the approach of [Cal+05] and introduce a relation for every basic concept, instead of only for the concept names.

We define the tuple such that the four sets are minimal and satisfy the r-completeness conditions. More precisely, we focus on those parts of the conditions which we can address without considering information that is not given (e.g., the mapping ι, which depends on the data). While the definition of components such as $Q_{\mathsf{R}[\mathcal{W}]}$ and $Q^{\neg}_{\mathsf{R}[\mathcal{W}]}$ is straightforward, the construction of the set R_{F} represents a challenge. This is because the focus on data complexity imposes special limits and Condition (C6) is rather intricate. Regarding data complexity, it would be critical to consider the entire set R_{F} and ABox $\mathcal{A}_{R_{\mathsf{F}}}$ within our rewriting. Recall that $\mathcal{A}_{R_{\mathsf{F}}}$ may contain auxiliary elements tailored to individual elements in $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$, which then would need to be considered explicitly within the rewriting—this is obviously impossible. To circumvent that, we first of all discern the assertions in R_{F} more fine-granularly, according to the kind of individual they address. In particular, the set R_{F} is considered to be the disjoint union of three sets $R_{\mathsf{F}|\mathsf{aux}}$, $R_{\mathsf{F}|\Phi}$, and $R_{\mathsf{F}|\mathsf{o}}$ (o for "other"), each containing only assertions on the individuals from $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$, $\mathsf{N}_{\mathsf{I}}(\Phi)$, and $\mathsf{N}_{\mathsf{I}}(\mathcal{K}) \setminus \mathsf{N}_{\mathsf{I}}(\Phi)$, respectively (i.e., next to the ones from $\mathsf{N}_{\mathsf{I}}^{\mathsf{tree}}$). For each of these sets, we can restrict our tests to parts of Condition (C6):

- $R_{\mathsf{F}|\mathsf{aux}}$ is constant and hence its size is not relevant. In particular, the elements of $\mathsf{N}_{\mathsf{I}}^{\mathsf{aux}}$ neither occur in the ABox type, nor in the input ABoxes, and can be uniquely associated to one of the CQs in Φ. For constructing $R_{\mathsf{F}|\mathsf{aux}}$ in line with Condition (C6), it is thus enough to focus on instantiations of the latter CQs, individually.

- The construction of $R_{\mathsf{F}|\mathsf{o}}$ is more involved. The issue with the size of the corresponding ABox is covered later in this section. To ensure that Condition (C6) is satisfied, we first model the derivation of all relevant (rigid) basic concept assertions, the consequences of the TKB, in our rewriting and then use them to derive those in $R_{\mathsf{F}|\mathsf{o}}$. Observe that the condition requires several ABoxes to be considered. Since the elements under consideration do not occur in Φ, we can however focus on the ABox type and the input ABoxes. That is, the rigid assertions represent (the relevant part of) the ABox type $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$; the derivation resolves transitivity and thus ensures that $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ is in line with Condition (C1). We define the set of these rigid assertions based on the below derivation as $\mathcal{B}_{\mathsf{R}|\mathsf{o}} := \mathcal{B}_{\mathsf{R}|\mathsf{o}}^{|\mathbb{B}_{\mathsf{R}}(\mathcal{O})|}$.

$$\begin{aligned}
\mathcal{B}_{\mathsf{R}|\mathsf{o}}^{0} &:= \emptyset,\\
\mathcal{B}_{\mathsf{R}|\mathsf{o}}^{j+1} &:= \{B(a) \mid B \in \mathbb{B}_{\mathsf{R}}(\mathcal{O}),\ a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \setminus \mathsf{N}_{\mathsf{I}}(\phi), \exists i.0 \leq i \leq n,\\
&\qquad\qquad \langle \mathcal{O}, \mathcal{B}_{\mathsf{R}|\mathsf{o}}^{j} \cup \mathcal{A}_i \rangle \models B(a)\}.
\end{aligned}$$

- The set $R_{\mathsf{F}|\Phi}$ is of constant size. For constructing it, we have to apply a derivation as above since the elements in $\mathsf{N}_{\mathsf{I}}(\Phi)$ might occur in the input ABoxes. Actually, these elements may occur in all the ABoxes regarded in Condition (C6), which thus all would have to be taken into account in the derivation. However, we can obviously not apply the mapping ι to define a fixed rewriting. This is why we assume a set $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N}_{\mathsf{I}}(\Phi)\}$ to be given instead of defining it. The test if $R_{\mathsf{F}|\Phi}$ satisfies Condition (C6), in dependence of $\mathcal{B}_\Phi$, is thus postponed to the actual application and evaluation of the rewriting.

These observations lead to the following definitions:

$$
\begin{aligned}
Q_{\mathsf{R}[\mathcal{W}]} &:= \{\alpha_j \in \mathcal{Q}_\Phi \mid W \in \mathcal{W},\ p_j \in W\},\\
Q^{\neg}_{\mathsf{R}[\mathcal{W}]} &:= \{\alpha_j \in \mathcal{Q}_\Phi \mid W \in \mathcal{W},\ p_j \notin W\},\\
R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]} &:= \{\exists S(a_y) \mid S \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}^-_{\mathsf{RR}},\ a_y \in \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}, \exists W_j \in \mathcal{W},\ \langle \mathcal{O}, \mathcal{A}_{Q_j}\rangle \models \exists S(a_y)\},\\
R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]} &:= \{\exists S(a) \in \mathcal{B}_\Phi \mid S \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}^-_{\mathsf{RR}}\},\\
R_{\mathsf{F}|\mathsf{o}} &:= \{\exists S(a) \mid S \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N}^-_{\mathsf{RR}},\ a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \setminus \mathsf{N}_{\mathsf{I}}(\Phi),\\
&\qquad\qquad \exists i.0 \leq i \leq n,\ \langle \mathcal{O}, \mathcal{B}_{\mathsf{R}|\mathsf{o}} \cup \mathcal{A}_i\rangle \models \exists S(a)\}.
\end{aligned}
$$

Based on these sets, we define the corresponding auxiliary ABoxes, as in Section 6.1.2:

- $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} := \bigcup_{\varphi \in Q_{\mathsf{R}[\mathcal{W}]}} \mathcal{C}_{\mathcal{O}}(\varphi)$.
- $\mathcal{A}_{Q_j}$ for $j \in [1,k]$ is defined in Section 6.1.2; for convenience, we sometimes write $\mathcal{A}_{Q_{W_j}}$ instead of $\mathcal{A}_{Q_j}$.
- $\mathcal{A}_{R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]}} := \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$; $\mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}}}, \mathcal{A}_{R_{\mathsf{F}|\Phi}}$, and $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ represent the ABoxes corresponding to the sets $R_{\mathsf{F}|\mathsf{aux}}$, $R_{\mathsf{F}|\Phi}$, and $R_{\mathsf{F}|\mathsf{o}}$, respectively, and are constructed in correspondence to the definition of $\mathcal{A}_{R_{\mathsf{F}}}$ in Section 6.1.2. Note that, while the former sets contain only flexible assertions, the ABoxes contain only rigid ones.
- $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ is then defined as the union of the set $\mathcal{A}^+_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, specified below, and all negative rigid role and basic concept assertions $\neg\alpha$ over $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$ for which we have $\alpha \notin \mathcal{A}^+_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. $\mathcal{B}_{\Phi|\mathsf{R}}$ denotes the set of rigid assertions in $\mathcal{B}_\Phi$.

$$
\begin{aligned}
\mathcal{A}^+_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} := \mathcal{B}_{\Phi|\mathsf{R}} \cup \mathcal{B}_{\mathsf{R}|\mathsf{o}} \cup \{\exists R(a) \mid R(a,e) \in \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}},\ a \in \mathsf{N}_{\mathsf{I}}(\Phi)\} \cup\\
\{R(a,b) \mid R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O}),\ a,b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}),\ R(a,b) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \text{ or}\\
\exists i.0 \leq i \leq n, \langle \mathcal{O}, \mathcal{A}_i\rangle \models R(a,b)\}
\end{aligned}
$$

  As mentioned above, we cannot assume the arbitrary set $\mathcal{B}_\Phi$ to be complete and to provide the part on the elements of $\mathsf{N}_{\mathsf{I}}(\Phi)$ which follows from the TKB, especially, regarding all the assertions to be contained in $\mathcal{A}_{\mathsf{R}}$. Since we test the sufficiency of $\mathcal{B}_\Phi$ when the rewriting is evaluated by using entailment tests, we then can however not detect rigid information missing in $\mathcal{B}_\Phi$ if it is implied by the flexible assertions. For that reason, we consider $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$ in the above construction and thus explicitly regard the consequences from the flexible assertions in $\mathcal{B}_\Phi$.

Observe that, apart from $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ and $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, which depend on the input ABoxes $\mathfrak{A}$, all of the ABoxes we defined are constant.

The below auxiliary lemma shows that $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ is, in a certain sense, closed regarding the data in the given ABox sequence.

Lemma 6.22 *For all $B \in \mathbb{B}_{\mathsf{R}}(\mathcal{O}), R \in \mathsf{N}_{\mathsf{RR}}(\mathcal{O})$ and $a,b \in \mathsf{N}_{\mathsf{I}}(\mathcal{K}) \setminus \mathsf{N}_{\mathsf{I}}(\Phi)$, we have:*

- *$B(a) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ iff there is an $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i\rangle \models B(a)$,*
- *$R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ iff there is an $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i\rangle \models R(a,b)$.*

Proof. ($\Rightarrow$) This direction is trivial. ($\Leftarrow$) We assume that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i \rangle \models B(a)$ holds for some $i \in [0,n]$. Observe that, if we have $\mathcal{B}^j_{\mathsf{R}|\mathsf{o}} = \mathcal{B}^{j+1}_{\mathsf{R}|\mathsf{o}}$ at some point, then we have $\mathcal{B}^j_{\mathsf{R}|\mathsf{o}} = \mathcal{B}^{j+l}_{\mathsf{R}|\mathsf{o}}$ for all $l \geq 0$. Hence, there must be some $j' \in [0, |\mathbb{B}_\mathsf{R}(\mathcal{O})|]$ such that $\mathcal{B}^{j'}_{\mathsf{R}|\mathsf{o}} = \mathcal{B}^{j'+l}_{\mathsf{R}|\mathsf{o}}$, for all $l \geq 0$, because every set $\mathcal{B}^{j+1}_{\mathsf{R}|\mathsf{o}}$ such that $\mathcal{B}^{j+1}_{\mathsf{R}|\mathsf{o}} \neq \mathcal{B}^j_{\mathsf{R}|\mathsf{o}}$ must contain at least one new assertion, there are only $|\mathbb{B}_\mathsf{R}(\mathcal{O})|$ relevant assertions per individual, and an assertion on a specific individual does not depend on assertions on other individuals by Lemma 2.14. Given $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i \rangle \models B(a)$ and the fact that only the assertions on the individual a (possibly contained in a tuple of individuals) in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ are relevant for the entailment by Lemmas 2.14 and 2.13, we can neglect all but the second and last conjuncts in the definition of $\mathcal{A}^+_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$; a cannot occur in these sets. In addition to the assertions in $\mathcal{B}_{\mathsf{R}|\mathsf{o}}$, we thus consider those entailed by some $\langle \mathcal{O}, \mathcal{A}_i \rangle$, $i \in [0,n]$. But the basic concept assertions corresponding to those role assertions have to be contained in $\mathcal{B}^1_{\mathsf{R}|\mathsf{o}}$ and hence in also in $\mathcal{B}_{\mathsf{R}|\mathsf{o}}$ by definition. This leads to $B(a) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ by the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$.

We second assume $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i \rangle \models R(a,b)$. By Definition 2.11 and Lemma 2.13, there is some $S \in \mathsf{N}^-_\mathsf{R}$ such that $S(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i$ and $\mathcal{O} \models S \sqsubseteq R$. If $S(a,b) \in \mathcal{A}_i$, then we have that $\langle \mathcal{O}, \mathcal{A}_i \rangle \models R(a,b)$, and hence $R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ by the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. Otherwise, that definition implies that there is a $j \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_j \rangle \models S(a,b)$, and hence also $\langle \mathcal{O}, \mathcal{A}_j \rangle \models R(a,b)$, which also shows that $R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. □

Given all the relevant ABoxes and additionally a mapping $\iota\colon [0,n] \to [1,k]$, we can specify the KBs in the focus of the r-completeness tests in Definition 6.8 for all $i \in [0, n+k]$, considering $\mathcal{A}_i := \emptyset$ for $i > n$:

$$\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} := \langle \mathcal{O}, \mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \rangle,$$

where

$$\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} := \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_{R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]}} \cup \mathcal{A}_i.$$

Before we continue regarding the rewriting, we show that we can use $\mathcal{B}_\Phi$ as intended.

Lemma 6.23 *For all $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ such that $\mathcal{W} = \{W_1,\dots,W_k\}$ and $\iota\colon [0,n] \to [1,k]$, there is an r-complete tuple w.r.t. $\mathcal{W}$ and ι iff there is a set $\mathcal{B}_\Phi$ such that the tuple*

$$(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}, Q_{\mathsf{R}[\mathcal{W}]}, Q^\neg_{\mathsf{R}[\mathcal{W}]}, R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]})$$

is r-complete w.r.t. $\mathcal{W}$ and ι.

Proof. ($\Leftarrow$) This direction is trivial. ($\Rightarrow$) We assume $(\mathcal{A}_\mathsf{R}, Q_\mathsf{R}, Q^\neg_\mathsf{R}, R_\mathsf{F})$ to be an r-complete tuple, define $\mathcal{B}_\Phi$ as follows, and show that $(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}, Q_{\mathsf{R}[\mathcal{W}]}, Q^\neg_{\mathsf{R}[\mathcal{W}]}, R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]})$ is r-complete as well:

$$\mathcal{B}_\Phi := \{B(a) \in \mathcal{A}_\mathsf{R} \cup R_\mathsf{F} \mid B \in \mathbb{B}(\mathcal{O}), a \in \mathsf{N}_\mathsf{I}(\phi)\}.$$

We focus on the conditions in Definition 6.8. Our tuple obviously satisfies Conditions (C3) and (C4) by construction.

Regarding Conditions (C1), (C2), and (C5), we describe a model of $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ that can be homomorphically embedded into the canonical interpretation of the consistent KB that exists for the given tuple, since it satisfies Condition (C1); we denote the latter KB by $\mathcal{K}^i_{\mathsf{R}}$. Observe that all positive assertions contained in one of the ABoxes of $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ must also be contained in $\mathcal{K}^i_{\mathsf{R}}$:

- $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \subseteq \mathcal{A}_{Q_{\mathsf{R}}}$ follows from the facts that both tuples satisfy Condition (C3) and $Q_{\mathsf{R}[\mathcal{W}]}$ is the minimal set satisfying that condition.
- $\mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}} \subseteq \mathcal{A}_{R_{\mathsf{F}}}$ is a consequence of the following observations. By definition, every $\exists S(b) \in R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$ is a consequence of a KB $\langle \mathcal{O}, \mathcal{A}_{Q_j} \rangle$, $W_j \in \mathcal{W}$. Since the given tuple satisfies Condition (C6) and we have $\iota(n+j) = j$ in that definition, the assertion is also contained in R_{F}. $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \subseteq \mathcal{A}_{R_{\mathsf{F}}}$ follows from the construction. Each $\exists S(b) \in R_{\mathsf{F}|\mathsf{o}}$ follows, by definition, from $\mathcal{B}_{\mathsf{R}|\mathsf{o}}$ together with some $\mathcal{A}_i$ (and $\mathcal{O}$). Since the given tuple satisfies Condition (C1), $\mathcal{A}_{\mathsf{R}}$ must contain all assertions in $\mathcal{B}_{\mathsf{R}|\mathsf{o}}$; since Condition (C6) is satisfied, the assertion is thus also contained in R_{F}.
- All positive assertions in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ have to be positive in $\mathcal{A}_{\mathsf{R}}$, too, by the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and the observations in the previous items. More precisely, we have $\mathcal{B}_{\Phi|\mathsf{R}} \subseteq \mathcal{A}_{\mathsf{R}}$, $\mathcal{B}_{\mathsf{R}|\mathsf{o}} \subseteq \mathcal{A}_{\mathsf{R}}$, and that all positive assertions in $\left(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \setminus \mathcal{B}_{\Phi|\mathsf{R}}\right) \setminus \mathcal{B}_{\mathsf{R}|\mathsf{o}}$ are implied by a KB $\langle \mathcal{O}, \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_i \rangle$, $i \in [0,n]$. Since the latter assertions occur in $\mathcal{K}^i_{\mathsf{R}}$, which is consistent by assumption, the ABox type $\mathcal{A}_{\mathsf{R}}$ also contains the latter rigid consequences.

Hence, any difference between $\mathcal{K}^i_{\mathsf{R}}$ and $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ (i.e., focusing on the assertions in $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and disregarding additional assertions in $\mathcal{K}^i_{\mathsf{R}}$) must be due to negative rigid assertions in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ that occur positively in $\mathcal{A}_{\mathsf{R}}$ (because $\mathcal{A}_{\mathsf{R}}$ is an ABox type) and may cause the inconsistency of $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. By providing a model for $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, we show that such assertions cannot exist. Since the given tuple satisfies Condition (C1) and $\mathcal{K}^i_{\mathsf{R}}$ contains all positive assertions occurring in $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, the KB $[\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}]^+$, obtained from $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ by dropping the negative assertions, is also consistent. We focus on the canonical interpretation $\mathcal{I}$ of that KB and show that it also satisfies $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. We consider negative role and basic concept assertions in $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$.

- Let $\neg R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. We prove $\mathcal{I} \not\models R(a,b)$ by contradiction, assuming that some of the ABoxes in $[\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}]^+$ contains a role assertion $S(a,b)$ such that $\mathcal{O} \models S \sqsubseteq R$. We thus consider the positive assertions in the ABoxes $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$, $\mathcal{A}_{Q_{\iota(i)}}$, $\mathcal{A}_{R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]}}$, and $\mathcal{A}_i$ and argue with the definitions of these ABoxes. Let S be rigid, first. We can disregard $\mathcal{A}_{Q_{\iota(i)}}$ and $\mathcal{A}_{R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]}}$, since all rigid assertions in the former ABox are also contained in $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$ and because the ABoxes $\mathcal{A}_{R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]}}$ consists of do not contain assertions on two elements of $\mathsf{N_I}(\mathcal{K})$. Further, (rigid) role assertions that contain only elements of $\mathsf{N_I}(\mathcal{K})$ are positively contained in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ if they occur in $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$ or are a consequence of a KB $\langle \mathcal{O}, \mathcal{A}_j \rangle$, $j \in [0,n]$. Together with the fact that $S(a,b) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$ implies $R(a,b) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$, this yields that the occurrence of $S(a,b)$ in one of the latter ABoxes causes $R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$.

Since $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ is an ABox type (i.e., only one $R(a,b)$ or $\neg R(a,b)$ is contained in it), that contradicts the assumption.

If S is flexible, $S(a,b)$ occurs in $\mathcal{A}_i$ or $\mathcal{A}_{Q_{\iota(i)}}$, which implies that $W_{\iota(i)} \in \mathcal{W}$. By the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$, based on $Q_{\mathsf{R}[\mathcal{W}]}$ and Definition 6.2, we then get $R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ or $R(a,b) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$, which also implies $R(a,b) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and thus a contradiction to the assumption.

- Let $\neg B(a) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. If $a \in \mathsf{N_I}(\Phi)$, this implies $\neg B(a) \in \mathcal{A}_{\mathsf{R}}$ by the definitions of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ based on $\mathcal{B}_{\Phi|\mathsf{R}}$ and $\mathcal{B}_\Phi$ based on $\mathcal{A}_{\mathsf{R}}$. Since $\mathcal{A}_{\mathsf{R}}$ is an ABox type and the given tuple satisfies Condition (C1), Lemma 2.12 yields $\mathcal{I}' \not\models B(a)$, assuming $\mathcal{I}'$ to be the canonical interpretation of $\mathcal{K}^i_{\mathsf{R}}$. By our above observation on the positive assertions in $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, this interpretation must also satisfy $[\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}]^+$. Hence, $B(a)$ cannot be a consequence of that KB, and Lemma 2.13 yields $\mathcal{I} \not\models B(a)$.

  For the case $a \notin \mathsf{N_I}(\Phi)$, we again proceed by contradiction and assume $\mathcal{I} \models B(a)$. Lemma 2.14 then yields that there are positive assertions about a in the ABoxes $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}} \cup \mathcal{A}_j$, $j \in [0,n]$, that together imply B; the other ABoxes do not contain assertions on such individuals. By that lemma, we can also disregard $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ since $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ contains the relevant assertions. Note that assertions in $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ only on elements of $\mathsf{N_I}(\mathcal{K})$ are always basic concept assertions. Then, Lemma 6.22 implies that we can actually focus on $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ alone and obtain $B(a) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. This contradicts the assumption since $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ is an ABox type.

Since $\mathcal{I}$ is a model of $[\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}]^+$ by Lemma 2.12 and we have shown that it satisfies all negative assertions in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, it is also a model of $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. Hence, our tuple satisfies Condition (C1).

If one of Conditions (C2) and (C5) is contradicted, then Lemma 2.13 yields that there is a homomorphism of the CQ that causes the contradiction into $\mathcal{I}$. Again, the above observation that the positive assertions contained in $\mathcal{K}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ must be contained in $\mathcal{K}^i_{\mathsf{R}}$ is important. By Definition 2.11 and the semantics, every such homomorphism into $\mathcal{I}$ is also a homomorphism into the canonical interpretation of the positive part of $\mathcal{K}^i_{\mathsf{R}}$. This contradicts the assumption that $\mathcal{K}^i_{\mathsf{R}}$ satisfies Conditions (C2) and (C5), again by Lemma 2.13.

It remains to consider Condition (C6), and we discern regarding the elements in focus. Observe that, w.r.t. the ABoxes considered in that condition, the individual names occurring in $R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$ can only occur within $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \bigcup_{W\in\mathcal{W}} \mathcal{A}_{Q_W}$, and those in $R_{\mathsf{F}|\mathsf{o}}$ only in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \bigcup_{0\le i\le n} \mathcal{A}_i$.

- We regard the assertions in $R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$. ($\Leftarrow$) For every $\exists S(a_y) \in R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$, and thus $a_y \in \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$, the definition of $R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$ directly yields that there is a $W_j \in \mathcal{W}$ such that $\langle \mathcal{O}, \mathcal{A}_{Q_j}\rangle \models \exists S(a_y)$. This solves the claim given that Condition (C6) considers ι to be such that $j = \iota(n+j)$.

  ($\Rightarrow$) If there is a $W \in \mathcal{W}$ such that $\langle \mathcal{O}, \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W}\rangle \models \exists S(a_y)$, $a_y \in \mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$, then Lemma 2.14 implies that $\exists S(a_y)$ can only follow from assertions involving a_y. But a_y can be associated to a unique query $\varphi_j \in \mathcal{Q}_\Phi$ that contains the variable y and corresponding ABox $\mathcal{A}_{\varphi_j}$; no other such ABoxes contains assertions on a_y. This implies $\varphi_j \in Q_{\mathsf{R}[\mathcal{W}]}$. By the definition of $Q_{\mathsf{R}[\mathcal{W}]}$, there is a $W' \in \mathcal{W}$ with $p_j \in W'$

and, in particular, $\mathcal{A}_{Q_{W'}}$ implies all assertions on a_y in $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$. This shows that $\exists S(a_y)$ already follows from $\mathcal{A}_{Q_{W'}}$ (and $\mathcal{O}$), which yields $\exists S(a_y) \in R_{\mathsf{F|aux}[\mathcal{W}]}$.

- We regard the assertions in $R_{\mathsf{F|o}}$. ($\Leftarrow$) If $\exists S(b) \in R_{\mathsf{F|o}}$, then there is an index $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{B}_{\mathsf{R|o}} \cup \mathcal{A}_i \rangle \models \exists S(b)$. By the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, we have $\mathcal{B}_{\mathsf{R|o}} \subseteq \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, and hence obtain that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i \rangle \models \exists S(b)$.

  ($\Rightarrow$) If $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i \rangle \models \exists S(b)$, then the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and Lemma 2.14 yield that the assertion follows from rigid assertions in $\mathcal{B}_{\mathsf{R|o}}$ and several rigid role assertions of the form $R(b,a)$, $R \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O})$, each of which follows from a KB $\langle \mathcal{O}, \mathcal{A}_j \rangle$, $j \in [0,n]$. But, for those rigid role assertions, we have that the corresponding rigid basic concept assertions $\exists R(b)$ are contained in $\mathcal{B}_{\mathsf{R|o}}$ by construction. Hence $\exists S(b)$ follows from $\mathcal{B}_{\mathsf{R|o}}$ (and $\mathcal{O}$), and is thus contained in $R_{\mathsf{F|o}}$, by the definition of that set.

- We regard the assertions in $R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$. Since the given tuple satisfies Condition (C6) and, by the definition of $R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$, R_{F} and $R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ coincide regarding $\mathsf{N_I}(\Phi)$, we have that there is an $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}} \cup \mathcal{A}_{Q_{\mathsf{R}}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \rangle \models \exists S(a)$ iff $\exists S(a) \in R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ for all $a \in \mathsf{N_I}(\Phi)$.

  ($\Rightarrow$) This direction then directly follows from the above observation that all positive assertions in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ occur in $\mathcal{A}_{\mathsf{R}}$ and $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \subseteq \mathcal{A}_{Q_{\mathsf{R}}}$.

  ($\Leftarrow$) Since $\mathcal{A}_{\mathsf{R}}$ is an ABox type, the fact that the given tuple satisfies Condition (C1) yields that all basic concept assertions that can be derived from assertions in $\mathcal{A}_{\mathsf{R}}$ or $\mathcal{A}_{Q_{\mathsf{R}}}$ are also positively contained in $\mathcal{A}_{\mathsf{R}}$; note that these ABoxes both contain only rigid assertions. Moreover, $\mathcal{B}_\Phi$ contains all these rigid basic concepts on elements of $\mathsf{N_I}(\Phi)$ which are contained in $\mathcal{A}_{\mathsf{R}}$, by definition. Hence, Lemma 2.14 yields that $\exists S(a) \in R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ implies that there is an $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \rangle \models \exists S(a)$.

Thus, Condition (C6) is also satisfied. □

In order to be able to specify rewritings, we next propose an approach to handle the critical size of the ABox $\mathcal{A}_{R_{\mathsf{F|o}}}$.

Introducing Prototypes

To untangle the knowledge captured in the ABox $\mathcal{A}_{R_{\mathsf{F|o}}}$ and separate it from the input data as much as possible, we introduce prototypes[11]. Specifically, instead of the names occurring in $\mathcal{A}_{R_{\mathsf{F|o}}}$, we consider prototypical, fresh individual names of the form $[S], a_{[S]S}, a_{[S]S\varrho}$, etc., meaning $[S]$ is used instead of a concrete individual name from $\mathsf{N_I}(\mathcal{K})$. We collect all these new names in the set $\mathsf{N}^{\mathsf{pro}}_{\mathsf{I}}$.

The ABox $\mathcal{A}_{\exists S}$ for all $S \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O}) \setminus \mathsf{N_{RR}}$ represents a prototypical version of an ABox $\mathcal{A}_{\exists S(b)}$ with $b \in \mathsf{N_I}(\mathcal{K})$ (see Section 6.1.2) and is obtained from $\mathcal{A}_{\exists S(b)}$ as follows, where $b \in \mathsf{N_I}(\mathcal{K})$ and $a_{b\varrho}, a_{b\varrho S} \in \mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$:

- every $R(b, a_{bS})$ is replaced by $R(b, a_{[S]S})$,

[11] Not to be confused with the prototypical elements in canonical interpretations.

- every $R(a_{b\varrho}, a_{b\varrho S})$ is replaced by $R(a_{[S]\varrho}, a_{[S]\varrho S})$,
- every $B(a_{b\varrho})$ is replaced by $B(a_{[S]\varrho})$.

For query answering regarding the adapted ABox, we propose the rewriting $\cdot^{\wp}$.

Definition 6.24 ($\cdot^{\wp}$) Let pro be a unary predicate that identifies exactly the elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. For a CQ $\varphi := \exists \vec{x}.\psi$, define the query $\varphi^{\wp} := \exists \vec{x}.\psi \wedge \psi_{\mathsf{filter}}$ with

$$\psi_{\mathsf{filter}} := \bigwedge_{\substack{R \in \mathsf{N}_\mathsf{R}^-, \\ R(s,t) \in \psi}} \left(\neg \mathsf{pro}(s) \wedge \mathsf{pro}(t) \rightarrow \bigwedge_{\substack{R' \in \mathsf{N}_\mathsf{R}^-, \\ R'(t,u) \in \psi}} (s = u \vee \mathsf{pro}(u)) \right).$$

The ABox $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ is defined as the union of the set $\bigcup_{\exists S(a) \in R_{\mathsf{F}|\mathsf{o}}} \mathcal{A}_{\exists S}$ and the set that, for each $a \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ occurring in the latter set, contains the assertion $\mathsf{pro}(a)$. ♢

The below lemma captures the intent of the rewriting.

Lemma 6.25 *Let φ be a CQ and $\mathcal{A} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ be one of the ABoxes $\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, where $i \in [0, n+k]$. Then, we have*

$$DB(\mathcal{A} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}) \models \varphi \text{ iff } DB(\mathcal{A} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}) \models \varphi^{\wp}.$$

Proof. Let $\varphi := \exists \vec{x}.\psi$. ($\Rightarrow$) Based on a homomorphism π of φ into $\mathrm{DB}(\mathcal{A} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}})$, we define a homomorphism π' of $\varphi^{\wp}$ into $\mathrm{DB}(\mathcal{A} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}})$. More precisely, π' corresponds to π but maps elements from $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$, which do not occur in $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$, to the corresponding prototypes, which exist in the domain of $\mathrm{DB}(\mathcal{A} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}})$ by the definition of $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$.

Let $R(s,t), S(t,u) \in \varphi; R, S \in \mathsf{N}_\mathsf{R}^-(\mathcal{O})$; $\pi'(s) \notin \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$; and $\pi'(t) \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, such that the precondition of ψ_{filter} evaluates to *true* under π'; otherwise, the implication clearly holds. Recall that, an ABox $\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ as considered (see the definition in the previous section) only via $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ refers to elements such as $\pi(t)$ of $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ that are associated to named individuals—here, $\pi(s)$—from $\mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$ (i.e., elements which are replaced by $\cdot^{\wp}$). By Definition 2.23, the assumption thus implies $R(\pi(s), \pi(t)), S(\pi(t), \pi(u)) \in \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$. Together with the assumption that $\pi'(s) \notin \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$; the fact that $\cdot^{\wp}$ replaces all of $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$, and only those, by prototypes; and $\pi'(s) = \pi(s)$, given by the definition of π', this means that $\pi(s) \notin \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$. Since $\pi(s)$ occurs in the ABox $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$, we hence have $\pi(s) \in \mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$. By the construction of $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ and the two role atoms contained in it, we thus must have $\pi(u) = \pi(s)$ or $\pi(u) \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$. This yields that either $\pi'(u) = \pi'(s)$ or $\pi'(u) \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ by our definition of π', which especially yields $\pi(s) = \pi'(s)$. Hence, π' satisfies $\varphi_{\mathsf{filter}}$.

($\Leftarrow$) Let π' be a homomorphism of $\varphi^{\wp}$ into $\mathrm{DB}(\mathcal{A} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}})$. We construct a homomorphism π of φ into $\mathrm{DB}(\mathcal{A} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}})$ and discern two cases. If π' does map to elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, but not to elements of $\mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$, then it maps only to elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. This is because atoms containing terms mapped to elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ can only be satisfied by $\mathrm{DB}(\mathcal{A} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}})$ through assertions in $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$, by Definition 2.23 and the fact that elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ do not occur in $\mathcal{A}$; $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ only contains individuals of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ and $\mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$; and

we assume φ to be connected. Furthermore, the elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ in $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ are connected in tree structures (see the definition of $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ in Section 6.1.2) and, for each such structure, there is at least one corresponding structure in $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ by the definition of $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$. We hence can define π to map to the corresponding tree elements in one such structure which correspond to the ones that π' maps to.

It remains to consider the case that π' maps to elements of both $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ and $\mathsf{N}_\mathsf{I}(\mathcal{K})\setminus\mathsf{N}_\mathsf{I}(\Phi)$.

First, we define π as π' w.r.t. all terms that are not mapped to elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. Note that $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ does not contain assertions which do *not* contain elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ (see the definition of $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$). By Definition 2.23, we thus have $\alpha \in \mathcal{A}$ for all atoms $\alpha \in \varphi$ where $\pi'(\alpha)$ does not contain elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. This particularly means that our definition of π is as required w.r.t. all those atoms.

For defining π on the terms not yet considered, we regard a term s in φ that is mapped to an element of $\mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$ and occurs in a role atom $R(s,t)$ together with a term t mapped to an element of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$; since we assume π' to map to elements of both $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ and $\mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$ and φ to be connected, such a role atom must exist. By Definition 2.23 and the fact that $\mathcal{A}$ does not contain elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, given by assumption, we have $R(\pi'(s),\pi'(t)) \in \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$.

Together, the definition of $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ and its dependence on $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ then yield that $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$ is based on an ABox $\mathcal{A}_{\exists S(\pi'(s))}$ such that $S \in \mathsf{N}_\mathsf{R}^-$ and $\exists S(\pi'(s)) \in R_{\mathsf{F}|\mathsf{o}}$; it contains the assertion $R(\pi'(s), a_{\pi'(s)S})$, $a_{\pi'(s)S} \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$; and, for all $R'(\pi'(s),\pi'(t)) \in \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$, we have $R'(\pi'(s), a_{\pi'(s)S}) \in \mathcal{A}_{\exists S(\pi'(s))}$. The latter follows from the observation that, apart from the ABox $\mathcal{A}_{\exists S(\pi'(s))}$, where $\pi'(s)$ is related to $a_{\pi'(s)S}$, no ABox $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ is based upon contains an assertion that relates $\pi'(s)$ to an element of the form a_{bS}, $b \in \mathsf{N}_\mathsf{I}(\mathcal{K})$. Thus, the introduction of prototypes yielding elements of the form $a_{[S]S}$ does not change the fact that all assertions in $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ on both $\pi'(s)$ and $a_{[S]S}$ must stem from $\in \mathcal{A}_{\exists S(\pi'(s))}$. Note that we also get $\pi'(t) = a_{[S]S}$. We define $\pi(t) = a_{\pi'(s)S}$. Since the filter conjunct is satisfied, we have that all role atoms $R'(t,u)$ in which t occurs contain only terms u that are either mapped to $\pi'(s)$ or to an element $\pi'(u) \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. The former kind of role atoms is satisfied by this definition of π by the above observation that all role assertions in $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ that contain both $\pi'(s)$ and $a_{[S]S}$ stem from $\mathcal{A}_{\exists S(\pi'(s))}$.

We regard a role atom of the other kind. Since this case is similar to the induction case, we abstract from the given information by assuming $\pi'(t) = a_{[S]\varrho}$, $\pi(t) = a_{b\varrho}$ to be defined, and $\exists S(b) \in R_{\mathsf{F}|\mathsf{o}}$. Again, by Definition 2.23 and the fact that $\mathcal{A}$ does not contain elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, by assumption, we have $R'(\pi'(t),\pi'(u)) \in \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$. By the definition of $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ based on ABoxes of the form $\mathcal{A}_{\exists S'}$, $S' \in \mathsf{N}_\mathsf{R}^-$, which do not overlap in the elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ they contain, we get that $R'(\pi'(t),\pi'(u)) \in \mathcal{A}_{\exists S}$ since the shape of $\pi'(t)$ indicates that it is contained in that ABox. By the same argument, we then get $R''(\pi'(t)),\pi'(u)) \in \mathcal{A}_{\exists S}$ for all $R''(\pi'(t),\pi'(u)) \in \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$. Given the way the prototypes are created and the definition of the ABox $\mathcal{A}_{\exists S}$, we can assume $\pi'(u)$ to be of the form $\pi'(u) = a_{[S]\varrho S'}$, $S' \in \mathsf{N}_\mathsf{R}^-$. From $\exists S(b) \in R_{\mathsf{F}|\mathsf{o}}$ and the construction of $\mathcal{A}_{\exists S(b)}$, we know that the element $a_{b\varrho S'} \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ exists and that $R''(\pi(t),\pi(u)) \in \mathcal{A}_{\exists S(b)}$ for all $R''(\pi'(t)),\pi'(u)) \in \mathcal{A}_{\exists S}$. We thus can define $\pi(u) = a_{\pi'(s)\varrho S'}$. □

If we apply this lemma in the following, we assume $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ to only contain the auxiliary elements used in $\mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$ and $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$, meaning that we disregard the auxiliary elements from $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$, which it contains according to the original definition. Also note that $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ does not contain elements from $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$.

6.3.2 Rewriting Knowledge Base Satisfiability and Query Answering

We next adapt the rewritings $q_{unsat(\mathcal{O})}$ and $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ to incorporate the knowledge from the auxiliary ABoxes. Observe, however, that for constructing auxiliary ABoxes such as $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$, we need to decide entailment w.r.t. different KBs already. Specifically, these tests include an index $i \in [0, n]$ as parameter and regard some of the other auxiliary ABoxes. For that reason, we first propose variants of $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ that similarly incorporate the given CQ φ and ontology but also the corresponding auxiliary ABoxes and that target our time-stamped database $\mathrm{TDB}(\mathfrak{A})$.

In the following, we assume that the CQs we consider only contain individual names from $\mathsf{N}_\mathsf{I}(\Phi)$.

First Variants of $\mathsf{PerfectRef}(\varphi, \mathcal{O})$

The formula $\mathsf{PRef}_{(\varphi,\mathcal{O})}(i)$ represents $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ regarding $\mathrm{TDB}(\mathfrak{A})$: Given a parameter $i \in [-1, n]$, it decides if φ is entailed by the atemporal KB $\langle \mathcal{O}, \mathcal{A}_i \rangle$. It is obtained from $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ by replacing all atoms $B(t)$ and $R(s, t)$ by $\mathsf{B}(t, i)$ and $\mathsf{R}(s, t, i)$, respectively. The correctness can be readily checked by considering the original semantics of the query (i.e., $\mathcal{O}$ is correctly included/rewritten w.r.t. the interpretation $\mathrm{DB}(\mathcal{A}_i)$) and the fact that $\mathrm{TDB}(\mathfrak{A})$ is defined such that $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{B}(a, i)$ iff $\mathrm{DB}(\mathcal{A}_i) \models B(a)$ for all $B \in \mathbb{B}(\mathcal{O})$ and $i \in [0, n]$, and correspondingly for all relations R in $\mathrm{TDB}(\mathfrak{A})$ where $R \in \mathsf{N}_\mathsf{R}(\mathcal{O})$.

Lemma 6.26 *For all $i \in [-1, n]$, $\langle \mathcal{O}, \mathcal{A}_i \rangle \models \varphi$ iff $\mathit{TDB}(\mathfrak{A}) \models \mathsf{PRef}_{(\varphi,\mathcal{O})}(i)$.* □

The formula $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^j_{\mathsf{R}|\mathsf{o}})}(i)$ represents $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ regarding $\mathrm{TDB}(\mathfrak{A})$ and $\mathcal{B}^j_{\mathsf{R}|\mathsf{o}}$ for all $j \in [0, \ell]$ with $\ell = |\mathbb{B}_\mathsf{R}(\mathcal{O})|$. We define:

- $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^0_{\mathsf{R}|\mathsf{o}})}(i) := \mathsf{PRef}_{(\varphi,\mathcal{O})}(i)$.
- $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^{j+1}_{\mathsf{R}|\mathsf{o}})}(i)$ for $j \in [1, \ell]$ is obtained from $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ by replacing all basic concept atoms $B(t)$, if flexible, by $\mathsf{B}(t, i)$ and otherwise by

$$\mathsf{B}(t, i) \vee \bigwedge_{a \in \mathsf{N}_\mathsf{I}(\Phi)} (t \neq a) \wedge \exists p.\mathsf{PRef}_{(B(t),\mathcal{O}|\mathcal{B}^j_{\mathsf{R}|\mathsf{o}})}(p),$$

  and all role atoms $R(s, t)$ by $\mathsf{R}(s, t, i)$.

Lemma 6.27 *For all $j \in [0, \ell]$ and $i \in [-1, n]$:*

$$\langle \mathcal{O}, \mathcal{B}^j_{\mathsf{R}|\mathsf{o}} \cup \mathcal{A}_i \rangle \models \varphi \text{ iff } \mathit{TDB}(\mathfrak{A}) \models \mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^j_{\mathsf{R}|\mathsf{o}})}(i).$$

Proof. The proof is by induction, based on the definition of the set $\mathcal{B}_{\mathsf{R|o}}$. For the base case, where $j = 0$, the definitions of $\mathcal{B}^0_{\mathsf{R|o}}$ and $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^0_{\mathsf{R|o}})}$ yield that the equivalence $\langle\mathcal{O}, \emptyset \cup \mathcal{A}_i\rangle \models \varphi$ iff $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{PRef}_{(\varphi,\mathcal{O})}(i)$ is to be shown, which holds by Lemma 6.26. We regard $j > 0$.

($\Rightarrow$) We assume $\langle\mathcal{O}, \mathcal{B}^j_{\mathsf{R|o}} \cup \mathcal{A}_i\rangle \models \varphi$ and thus get $\mathrm{DB}(\mathcal{B}^j_{\mathsf{R|o}} \cup \mathcal{A}_i) \models \mathsf{PerfectRef}(\varphi, \mathcal{O})$ by Lemma 2.24. That is, there is a homomorphism π of $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ into $\mathrm{DB}(\mathcal{B}^j_{\mathsf{R|o}} \cup \mathcal{A}_i)$ and hence a CQ φ' in the UCQ $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ such that, for all atoms α in φ', $\pi(\alpha) \in \mathcal{B}^j_{\mathsf{R|o}} \cup \mathcal{A}_i$, by the query semantics and the definition of $\mathrm{DB}(\mathcal{B}^j_{\mathsf{R|o}} \cup \mathcal{A}_i)$ (see Definition 2.23). For all these atoms, we show that π is also a homomorphism of the corresponding replacement into $\mathrm{TDB}(\mathfrak{A})$.[12]

- If α is a flexible basic concept atom $B(t)$ or a role atom $R(s, t)$, then $\pi(\alpha)$ cannot be contained in the set $\mathcal{B}^j_{\mathsf{R|o}}$ containing only rigid basic concepts, by definition; hence, we get $\pi(\alpha) \in \mathcal{A}_i$. The definition of $\mathrm{TDB}(\mathfrak{A})$ (see Definition 6.21) then implies that B^{DB} contains the tuple $(\pi(t), i)$ or that R^{DB} contains the tuple $(\pi(s), \pi(t), i)$, respectively. Thus, π is as required.

- Let α be a rigid basic concept atom $B(t)$. If $\pi(\alpha) \in \mathcal{B}^j_{\mathsf{R|o}}$, then the definition of $\mathcal{B}^j_{\mathsf{R|o}}$, especially the fact that $\mathcal{B}^j_{\mathsf{R|o}}$ contains only individual names from $\mathsf{N_I}(\mathcal{K}) \setminus \mathsf{N_I}(\Phi)$, yields that $\pi(t) \notin \mathsf{N_I}(\Phi)$ and that there is a $k \in [0, n]$ such that $\langle\mathcal{O}, \mathcal{B}^{j-1}_{\mathsf{R|o}} \cup \mathcal{A}_k\rangle \models \pi(\alpha)$. By the induction hypothesis, the latter implies $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{PRef}_{(\pi(\alpha),\mathcal{O}|\mathcal{B}^{j-1}_{\mathsf{R|o}})}(k)$, which together with the former yields that π is also a homomorphism of the replacement into $\mathrm{TDB}(\mathfrak{A})$ and thus as required.

  If $\pi(\alpha) \in \mathcal{A}_i$, then B^{DB} contains the tuple $(\pi(t), i)$ by the definition of $\mathrm{TDB}(\mathfrak{A})$.

($\Leftarrow$) We argue correspondingly and consider a given homomorphism π and a satisfied disjunct of $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^j_{\mathsf{R|o}})}(i)$, which itself is a conjunction and, before our replacements, originally had been a CQ obtained from $\mathsf{PerfectRef}(\varphi, \mathcal{O})$. We consider an arbitrary conjunct α in that conjunction and show that π is also a homomorphism of the atom that has been replaced by the conjunct into $\mathrm{DB}(\mathcal{B}^j_{\mathsf{R|o}} \cup \mathcal{A}_i)$.

- If α has replaced a role atom $R(s, t)$ or flexible basic concept atom $B(t)$, then it is of the form $\mathsf{R}(s, t, i)$ or, respectively, $\mathsf{B}(t, i)$. By the assumption that $\mathrm{TDB}(\mathfrak{A}) \models \pi(\alpha)$ and the definition of $\mathrm{TDB}(\mathfrak{A})$, we then directly get that $R(\pi(s), \pi(t)) \in \mathcal{A}_i$ or, respectively, $B(\pi(t)) \in \mathcal{A}_i$. Hence, π is as required w.r.t. the original atom and the database.

- Let α be the replacement of a rigid basic concept atom $B(t)$. If the first disjunct $\mathsf{B}(t, i)$ in α is satisfied, we can argument as in the previous case. Otherwise, we have $\mathrm{TDB}(\mathfrak{A}) \models \exists p.\mathsf{PRef}_{(B(\pi(t)),\mathcal{O}|\mathcal{B}^{j-1}_{\mathsf{R|o}})}(p)$ and $\pi(t) \notin \mathsf{N_I}(\Phi)$. But this yields $\langle\mathcal{O}, \mathcal{B}^{j-1}_{\mathsf{R|o}} \cup \mathcal{A}_p\rangle \models B(\pi(t))$ by the induction hypothesis. Then, we have

[12] We assume the notion of homomorphism to be extended to the replacements, which may be nested disjunctions of rewritings (i.e., $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{B}^{j+1}_{\mathsf{R|o}})}(i)$ is a disjunction whose disjuncts are conjunctions where each conjunct my be an inequality, a disjunction or, in turn, be recursively defined as to be of that form) and equality assertions in the obvious way.

$B(\pi(t)) \in \mathcal{B}^j_{\mathsf{R|o}}$, by the definition of $\mathcal{B}^j_{\mathsf{R|o}}$, and hence again get that π is as required. □

The formula $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(i)$ represents $\mathsf{PerfectRef}(\varphi,\mathcal{O})$ regarding $\mathrm{TDB}(\mathfrak{A})$ and $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. It is obtained from $\mathsf{PerfectRef}(\varphi,\mathcal{O})$ by replacing

- all basic atoms $B(t)$, if flexible, by $\mathsf{B}(t,i)$ and otherwise by

$$\left(\bigvee_{B(a)\in\mathcal{B}_{\Phi|\mathsf{R}}} (t=a)\right) \vee \mathsf{PRef}_{(B(t),\mathcal{O}|\mathcal{B}_{\mathsf{R|o}})}(i) \vee \left(\bigvee_{\substack{B=\exists R,a\in\mathsf{N_I}(\mathcal{K}),\\ R(a,e)\in\mathcal{A}_{R_{\mathsf{F}|\Phi}[\mathcal{B}_\Phi]}}} (t=a)\right);$$

- all role atoms $R(s,t)$, if flexible, by $\mathsf{R}(s,t,i)$ and otherwise by

$$\exists p.\mathsf{PRef}_{(R(s,t),\mathcal{O})}(p) \vee \left(\bigvee_{R(a,b)\in\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}} (t=a)\wedge(t=b)\right).$$

Given the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ and the rewritings introduced above, it can be readily checked that the adapted query decides if φ is entailed by the KB $\langle\mathcal{O},\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i\rangle$.

Lemma 6.28 *$\langle\mathcal{O},\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i\rangle \models \varphi$ iff $TDB(\mathfrak{A}) \models \mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(i)$.*

Proof. Since we construct the rewriting based on $\mathsf{PerfectRef}(\varphi,\mathcal{O})$, Lemma 2.24 yields $\langle\mathcal{O},\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i\rangle \models \varphi$ iff $\mathrm{DB}(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i) \models \mathsf{PerfectRef}(\varphi,\mathcal{O})$.

($\Rightarrow$) We assume $\langle\mathcal{O},\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i\rangle \models \varphi$, which hence means that there is a disjunct φ' in the UCQ $\mathsf{PerfectRef}(\varphi,\mathcal{O})$ and a homomorphism π of φ' into $\mathrm{DB}(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i)$. That is, for all atoms α in φ', we have $\pi(\alpha)\in\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i$ by Definition 2.23.

- If α is flexible, then we have $\pi(\alpha)\in\mathcal{A}_i$ since $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ contains only rigid assertions. But then, π also fits our rewriting for that case, which addresses the corresponding relation in $\mathrm{TDB}(\mathfrak{A})$.
- If α is rigid and $\pi(\alpha)\in\mathcal{A}_i$, then the definition of $\mathcal{B}_{\mathsf{R|o}}$ yields $\pi(\alpha)\in\mathcal{B}_{\mathsf{R|o}}$ if we consider a basic concept atom. Thus, $\mathrm{TDB}(\mathfrak{A})$ satisfies either $\mathsf{PRef}_{(\pi(\alpha),\mathcal{O}|\mathcal{B}_{\mathsf{R|o}})}(i)$ or $\mathsf{PRef}_{(\pi(\alpha),\mathcal{O})}(i)$, depending on the kind of atom, by Lemmas 6.26 and 6.27. Hence, our rewriting is as required for this case.

  For the remaining case where $\pi(\alpha)\in\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, it can readily be checked that the rewriting covers all parts referenced in the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, again using Lemmas 6.26 and 6.27.

($\Leftarrow$) We argument correspondingly and consider a given homomorphism π and a satisfied disjunct in the UCQ $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(i)$, which itself is a conjunction. That is, it is a CQ obtained from $\mathsf{PerfectRef}(\varphi,\mathcal{O})$ where the atoms are replaced according to our above construction. We focus on an arbitrary such replacement for an atom α and show that we have $\pi(\alpha)\in\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\cup\mathcal{A}_i$. By the definition of the replacements, the replacement must be a disjunction, and at least one of its disjuncts must be satisfied

under π. If this disjunct refers to an assertion contained in $\mathcal{B}_{\Phi|\mathsf{R}}, \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$, or $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$, then we have $\pi(\alpha) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ by the definition of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. If the disjunct refers to the rewriting $\mathsf{PRef}_{(\alpha,\mathcal{O})}$, then Lemma 6.26 yields $\pi(\alpha) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. In case it refers to the rewriting $\mathsf{PRef}_{(\alpha,\mathcal{O}|\mathcal{B}_{\mathsf{R}|\mathsf{o}})}$, then α is a rigid basic concept atom of the form $B(t)$, and Lemma 6.27 yields $\langle \mathcal{O}, \mathcal{B}_{\mathsf{R}|\mathsf{o}} \cup \mathcal{A}_i\rangle \models B(\pi(t))$. Because of $\mathcal{B}_{\mathsf{R}|\mathsf{o}} \subseteq \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, we thus also have $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i\rangle \models B(\pi(t))$. For that reason, Lemma 6.22 leads to $B(\pi(t)) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. Thus, π is also a homomorphism of $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ into $\mathrm{DB}(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i)$ by Definition 2.23. □

The Final Versions of $q_{unsat(\mathcal{O})}$ and $\mathsf{PerfectRef}(\varphi, \mathcal{O})$

We finally come to the rewritings we target. Recall that, for checking the conditions for r-completeness (see Definition 6.8), we need to decide KB consistency and CQ entailment by including, next to the input ABoxes $\mathfrak{A} = (\mathcal{A}_i)_{0\leq i\leq n}$, several auxiliary ABoxes based on the tuple defined at the beginning of this section. Specifically, we focus on the KBs $\langle \mathcal{O}, \mathcal{A}^{i\,\wp}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}\rangle$ with $i \in [0, n]$ and

$$\mathcal{A}^{i\,\wp}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} := \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}} \cup \mathcal{A}_i.$$

Observe that the ABoxes $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}, \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}, \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}, \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$, and $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ are constant, whereas $\mathcal{A}_{Q_{\iota(i)}}$ and $\mathcal{A}_i$ depend on the considered time point i. Since the auxiliary ABoxes are not part of the input (of the r-satisfiability problem), we in the following adapt the FO formulas $q_{unsat(\mathcal{O})}$ and $\mathsf{PerfectRef}(\varphi, \mathcal{O})$ such that the two problems can be solved by evaluating the respective formula over the input ABoxes alone (i.e., also without the ontology), which are captured by $\mathrm{TDB}(\mathfrak{A})$.

For all sets $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$, $W \in \mathcal{W}$, and $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N_I}(\Phi)\}$, which are constant, we create the rewritings $q_{unsat(\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$ and $\mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$, based on $q^{\wp}_{unsat(\mathcal{O})}$ and $\mathsf{PerfectRef}(\varphi, \mathcal{O})^{\wp}$, respectively;[13] $\cdot^{\wp}$ represents a function that applies the rewriting $\cdot^{\wp}$ to every CQ in the given UCQ. That is, we adopt the approach proposed by Lemma 6.25; in line with this, we also consider $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ instead of $\mathcal{A}_{R_{\mathsf{F}|\mathsf{o}}}$. As in the previous section, we provide replacements for the atoms in these two queries.

Since the database addressed by these queries contains only the individuals occurring in the input ABoxes, but we want to include auxiliary elements from $\mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$, $\mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$, and $\mathsf{N}^{\mathsf{pro}}_{\mathsf{I}}$, we especially have to consider the quantifiers, which quantify only over the individuals in the original database. Note that this issue is not relevant regarding the rewritings we proposed in the previous section because we assume all individual names occurring in Φ to also occur in the input ABoxes and, in that section, focus on auxiliary ABoxes that do not contain names from $\mathsf{N}^{\mathsf{aux}}_{\mathsf{I}}$, $\mathsf{N}^{\mathsf{tree}}_{\mathsf{I}}$, and $\mathsf{N}^{\mathsf{pro}}_{\mathsf{I}}$. To this end, we now consider each disjunct

$$q = \exists x_1, \dots, x_\ell.\varphi(x_1, \dots, x_\ell) \wedge \varphi_{\mathsf{filter}}(x_1, \dots, x_\ell)$$

contained in the original queries—because of the filter condition included by $\cdot^{\wp}$, we do not have UCQs anymore—, where $\varphi(x_1, \dots, x_\ell)$ is a conjunction of atoms. The idea

[13] Note that we here use the subscript part $\mathcal{W}, W, \mathcal{B}_\Phi$ to describe the dependence of the rewriting on the given parameters, whereas we have used the additionally included ABoxes above. For brevity, we here do not use $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \cup \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$.

is to duplicate q several times such that we have 2^ℓ versions $q_0, \ldots, q_{2^\ell-1}$ of it, and to augment the quantification of variables to consider also the elements in $\mathsf{N}_\mathsf{I}^{\mathsf{aux}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{tree}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, a constant number. We then replace q by the disjunction $q_0 \vee \cdots \vee q_{2^\ell-1}$.

Formally, for all $j \in [0, \ell-1]$ and $k \in [0, 2^\ell - 1]$, we consider the quantified variable x_j in q_k with

$$k = b_0 * 2^0 + \ldots + b_j * 2^j + \ldots + b_{\ell-1} * 2^{\ell-1}$$

and drop the quantification if $b_j = 0$. We then define

$$q_k := \exists' x_1 \ldots . \exists' x_\ell . \mathsf{rep}_{(\varphi \wedge \varphi_{\mathsf{filter}} | \mathcal{W}, W, \mathcal{B}_\Phi)}(i)$$

where $\exists' x_j$ stands for $\exists x_j$ if $b_j = 1$, and for $\bigvee_{a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{tree}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{pro}}}$ if $b_j = 0$. For every $b \in \{0,1\}$, we denote by $Q_b(k)$ the set $\{x_j \mid 0 \leq j \leq \ell - 1, b_j = b\}$. The formula $\mathsf{rep}_{(\varphi \wedge \varphi_{\mathsf{filter}} | \mathcal{W}, W, \mathcal{B}_\Phi)}(i)$ is constructed by replacing every atom α in $\varphi \wedge \varphi_{\mathsf{filter}}$ by $\mathsf{rep}_{(\alpha | \mathcal{W}, W, \mathcal{B}_\Phi)}(i)$, in dependence of the form of α:

- For all $t \in \mathsf{N}_\mathsf{I}(q) \cup \mathsf{N}_\mathsf{V}(q)$:

$$\mathsf{rep}_{(\mathsf{pro}(t) | \mathcal{W}, W, \mathcal{B}_\Phi)}(i) := \begin{cases} \mathit{true} & \text{if } t \in Q_0(k), t = x_j, a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}, \\ \mathit{false} & \text{otherwise.} \end{cases}$$

- For all $s, t \in \mathsf{N}_\mathsf{I}(q) \cup \mathsf{N}_\mathsf{V}(q)$:

$$\mathsf{rep}_{(s=t | \mathcal{W}, W, \mathcal{B}_\Phi)}(i) := \begin{cases} s = t & \text{if } s, t \in Q_1(k) \cup \mathsf{N}_\mathsf{I}(q), \\ \mathit{true} & \text{if } s, t \in Q_0(k),\ s = x_j,\ t = x_{j'},\ a_j = a_{j'}, \\ \mathit{false} & \text{otherwise.} \end{cases}$$

- The remaining kinds of atoms are covered in Figures 6.1 and 6.2.

Observe that neither the two above case distinctions nor the tables conditions referencing $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}, \mathcal{A}_{Q_W}, \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$, and $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$ depend on the input ABoxes. Furthermore, the definitions basically consider two different cases, depending on whether the atom contains a variable which gets unquantified by our adaptation of the query. If this is the case, then it has to be mapped to an auxiliary element, and hence corresponding assertions (i.e., for the atom under consideration) can only occur in $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$; otherwise, the input ABoxes and $\mathcal{A}_{\mathsf{R}[\mathcal{W}, \mathcal{B}_\Phi]}$ are also taken into account—by using the query $\mathsf{PRef}_{(\varphi, \mathcal{O} | \mathcal{A}_{\mathsf{R}[\mathcal{W}, \mathcal{B}_\Phi]})}$. The variables that get unquantified by our adaptation and are associated to an element from $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ represent however an exception. For them, the input ABoxes have to be considered to ensure that corresponding prototypical elements actually occur in $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$. Also recall that the same definitions apply for $i = -1$, where we assume the considered ABox $\mathcal{A}_i$ to be empty.

The next lemma establishes the correctness of our translation of the original UCQs over $\mathcal{K}^i_{\mathsf{R}[\mathcal{W}, \mathcal{B}_\Phi]}$ into FO formulas over TDB($\mathfrak{A}$).

Lemma 6.29 *For all CQs φ over concept and role names in $\mathcal{O}$, $\mathcal{W} \subseteq 2^{\{p_1, \ldots, p_m\}}$, worlds $W \in \mathcal{W}$, sets $\mathcal{B}_\Phi \subseteq \bigcup_{\substack{B \in \mathbb{B}(\mathcal{O}), \\ a \in \mathsf{N}_\mathsf{I}(\Phi)}} \{B(a)\}$, and indexes $i \in [-1, n]$, we have:*

Conditions	$\mathsf{rep}_{(B(t)\mid\mathcal{W},W,\mathcal{B}_\Phi)}(i)$
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}}$, $B(a_j) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W}$	*true*
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$, $B(a_j) \in \mathcal{A}_{R_{\mathsf{F}\mid\mathsf{aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}\mid\Phi[\mathcal{B}_\Phi]}}$	*true*
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, $\langle\emptyset, \mathcal{A}_{\exists S}\rangle \models B(a_j)$, and $a_j = a_{[S]S_\varrho}$	$\exists x.\left(\bigwedge_{a\in\mathsf{N}_\mathsf{I}(\Phi)}(x \neq a)\right) \wedge$ $\exists p.\mathsf{PRef}_{(\exists y.S(x,y),\mathcal{O}\mid\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(p)$
Otherwise	*false*

(a) $t \in Q_0(k)$ $(t = x_j)$

Conditions	$\mathsf{rep}_{(B(t)\mid\mathcal{W},W,\mathcal{B}_\Phi)}(i)$
$B \notin \mathbb{B}_\mathsf{R}(\mathcal{O})$	$\left(\bigvee_{\substack{B(a)\in\mathcal{A}_{Q_W},\\ a\in\mathsf{N}_\mathsf{I}(\Phi)}}(t = a)\right) \vee \mathsf{B}(t,i)$
Otherwise	$\left(\bigvee_{\substack{B(a)\in\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}},\\ a\in\mathsf{N}_\mathsf{I}(\Phi)}}(t = a)\right) \vee \mathsf{PRef}_{(B(t),\mathcal{O}\mid\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(i)$

(b) $t \in Q_1(k) \cup \mathsf{N}_\mathsf{I}(q)$

Figure 6.1: The replacement definition for all $B \in \mathbb{B}(\mathcal{O})$ and $t \in \mathsf{N}_\mathsf{I}(q) \cup \mathsf{N}_\mathsf{V}(q)$.

Conditions	$\mathsf{rep}_{(R(s,t)\mid\mathcal{W},W,\mathcal{B}_\Phi)}(i)$
$a_j, a_{j'} \notin \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, $R(a_j, a_{j'}) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_{R_{\mathsf{F}\mid\mathsf{aux}[\mathcal{W}]}}$	*true*
$a_j, a_{j'} \notin \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, $R(a_j, a_{j'}) \in \mathcal{A}_{R_{\mathsf{F}\mid\Phi[\mathcal{B}_\Phi]}}$	*true*
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, $R(a_j, a_{j'}) \in \mathcal{A}_{\exists S}$ and $a_j = a_{[S]\varrho}$	$\exists x.\Big(\bigwedge_{a\in\mathsf{N}_\mathsf{I}(\Phi)} (x \neq a)\Big) \wedge$ $\exists p.\mathsf{PRef}_{(\exists y.S(x,y),\mathcal{O}\mid\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(p)$
Otherwise	*false*

(a) $s, t \in Q_0(k)$ ($s = x_j$, $t = x_{j'}$)

Conditions	$\mathsf{rep}_{(R(s,t)\mid\mathcal{W},W,\mathcal{B}_\Phi)}(i)$
$R(a_j, t) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W}$	*true*
$R(a_j, t) \in \mathcal{A}_{R_{\mathsf{F}\mid\Phi[\mathcal{B}_\Phi]}}$	*true*
Otherwise	*false*

(b) $s \in Q_0(k)$, $t \in \mathsf{N}_\mathsf{I}(q)$ ($s = x_j$)

Conditions	$\mathsf{rep}_{(R(s,t)\mid\mathcal{W},W,\mathcal{B}_\Phi)}(i)$
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}}$	$\bigvee_{R(a_j,b)\in\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}\cup\mathcal{A}_{Q_W}, b\in\mathsf{N}_\mathsf{I}(\Phi)} (t = b)$
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$	$\bigvee_{R(a_j,b)\in\mathcal{A}_{R_{\mathsf{F}\mid\Phi[\mathcal{B}_\Phi]}}, b\in\mathsf{N}_\mathsf{I}(\Phi)} (t = b)$
$a_j \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, $R(a_j, [S]) \in \mathcal{A}_{\exists S}$, and $a_j = a_{[S]S}$	$\bigwedge_{b\in\mathsf{N}_\mathsf{I}(\Phi)} (t \neq b) \wedge$ $\exists p.\mathsf{PRef}_{(\exists y.S(t,y),\mathcal{O}\mid\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(p)$
Otherwise	*false*

(c) $s \in Q_0(k)$, $t \in Q_1(k)$ ($s = x_j$)

Conditions	$\mathsf{rep}_{(R(s,t)\mid\mathcal{W},W,\mathcal{B}_\Phi)}(i)$
$R \notin \mathsf{N}_{\mathsf{RR}}$	$\Big(\bigvee_{\substack{R(a,b)\in\mathcal{A}_{Q_W},\\ a,b\in\mathsf{N}_\mathsf{I}(\Phi)}} (s = a) \wedge (t = b)\Big) \vee \mathsf{R}(s,t,i)$
Otherwise	$\Big(\bigvee_{\substack{R(a,b)\in\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}},\\ a,b\in\mathsf{N}_\mathsf{I}(\Phi)}} (s = a) \wedge (t = b)\Big) \vee \mathsf{PRef}_{(R(s,t),\mathcal{O}\mid\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(i)$

(d) $s, t \in Q_1(k) \cup \mathsf{N}_\mathsf{I}(q)$

Figure 6.2: The replacement definition for all $R \in \mathsf{N}_\mathsf{R}^-(\mathcal{O})$ and $s, t \in \mathsf{N}_\mathsf{I}(q) \cup \mathsf{N}_\mathsf{V}(q)$.

- $DB(\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}) \models q_{unsat(\mathcal{O})}$ *iff* $TDB(\mathfrak{A}) \models q_{unsat(\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$.
- $DB(\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}) \models \mathsf{PerfectRef}(\varphi,\mathcal{O})$ *iff* $TDB(\mathfrak{A}) \models \mathsf{PRef}_{(\varphi,\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$.

Proof. ($\Rightarrow$) We assume $\mathrm{DB}(\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}) \models q'$, where q' is one of the two UCQs, and show that $\mathrm{TDB}(\mathfrak{A})$ satisfies one of the queries in the disjunction representing the corresponding rewriting. By the definition of $\cdot^\wp$, the semantics of disjunction, and Lemma 6.25, we can equivalently assume $\mathrm{DB}(\mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}{}^\wp) \models (q')^\wp$. Hence, there is a homomorphism π of some query

$$q^\wp = \exists x_1,\ldots,x_\ell.\varphi(x_1,\ldots,x_\ell) \wedge \varphi_{\mathsf{filter}}(x_1,\ldots,x_\ell)$$

contained in $(q')^\wp$ into that database, where $\varphi(x_1,\ldots,x_\ell)$ is a conjunction of atoms. By definition, our translation of $(q')^\wp$ further contains a query q_k that is an adaptation of $q^\wp$ and such that

$$k = b_0 * 2^0 + \ldots + b_j * 2^j + \ldots + b_{\ell-1} * 2^{\ell-1}$$

where $b_j = 0$ iff $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{tree}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$ for all $j \in [0,\ell-1]$.

We show that, if it is restricted to $\mathsf{N_V}(q_k) \cup \mathsf{N_I}(q_k)$, π is a homomorphism of q_k into $\mathrm{TDB}(\mathfrak{A})$.[14] For that, we consider all conjuncts α of $q^\wp$ and replacements $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$ introduced by our adaptation and prove that $\pi(\alpha)$ is satisfied by the database iff $\pi(\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i))$ is satisfied by $\mathrm{TDB}(\mathfrak{A})$. Observe that this approach focusing on those atoms in $q^\wp$ that contain quantified variables or individual names, and the corresponding replacements, is correct since the queries are Boolean. Furthermore, since the case that $\pi(\alpha)$ is not satisfied by the database can only occur w.r.t. the filtering conjunct, it actually suffices to show implication regarding all atoms with predicates different from pro. By the assumption, the database satisfies the assertion $\pi(\alpha)$, obtained from α by replacing the variable(s) x in α by $\pi(x)$. By Definition 2.23, we thus get that $\pi(\alpha) \in \mathcal{A}^i_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}{}^\wp$.

Regarding the case that $\alpha = \mathsf{pro}(x_j)$, the correspondence of the definitions of the predicate pro and $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$ directly yields the equivalence. That is, $\pi(\alpha)$ evaluates to *true* iff $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i) = \mathit{true}$, which means that the restriction of π (trivially) fits in this regard. Note that the shape of these replacements yields that the adaptation of $\varphi_{\mathsf{filter}}$ is basically a conjunction of equality statements (see Definition 6.24).

Regarding the other kinds of atoms, we focus on the types of the mapped elements. As mentioned above, the definitions of the replacement formulas overall depend on whether the regarded atom contains a variable x_j, $j \in [0,\ell-1]$, such that $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{tree}} \cup \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. For that reason, we assume α to be an arbitrary atom containing such a variable x_j, in all but the last of the below cases, where we show that the above mentioned implication holds. The scheme of these proofs is the same: the kinds of the considered elements (i.e., regarding all elements π maps a term of α to) constrain the ABoxes to be considered, and the definition of $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$ is based on exactly those ABoxes.

- Let $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}} \cup \mathsf{N_I}(\mathcal{A}_{R_{\mathsf{F}|_{\mathsf{aux}[\mathcal{W}]}}})$. That is, we also regard elements from $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ of the form $a_{a_x\varrho}$, where $a_x \in \mathsf{N}_\mathsf{I}^{\mathsf{aux}}$. We hence must have $\pi(\alpha) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_{R_{\mathsf{F}|_{\mathsf{aux}[\mathcal{W}]}}}$

[14] Note that we have not defined the notion of homomorphism w.r.t. disjunction ($\vee$) and (in)equality predicates in Definition 3.5, but the corresponding extension should be obvious.

because the input ABoxes and $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ only regard elements of $\mathsf{N}_\mathsf{I}(\mathcal{K})$ and neither $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$ nor $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ contain elements of this kind. For the case that $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$, we obtain $\pi(\alpha) \in \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$ by analogous arguments and the fact that neither $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$ nor $\mathcal{A}_{Q_W}$ contains elements from $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$.

If α does not contain a variable that is quantified within q_k, then we directly get $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i) = \mathit{true}$, which is trivially satisfied by π. For basic concept atoms, this holds because we only have to consider $\pi(x_j)$ and $\mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$ does not contain basic concept assertions on elements of $\mathsf{N}_\mathsf{I}^{\mathsf{aux}}$. Regarding role atoms, observe that π cannot map to elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, since they do not occur within $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$, and that $\mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$ does not contain role assertions with individuals from $\mathsf{N}_\mathsf{I}(\mathcal{K})$. The latter is relevant regarding the case where $t \in \mathsf{N}_\mathsf{I}(q)$ (in the definition of the replacement), since we have $\mathsf{N}_\mathsf{I}(q_k) \subseteq \mathsf{N}_\mathsf{I}(\Phi)$ and assume $\mathsf{N}_\mathsf{I}(\Phi) \subseteq \mathsf{N}_\mathsf{I}(\mathcal{K})$.

If α next to x_j contains a variable y that is quantified within q_k—which means we regard the case $s \in Q_0(k), t \in Q_1(k)$—, then we have $\pi(y) \in \mathsf{N}_\mathsf{I}(\mathcal{K})$, by the semantics, and that α is a role atom. Let $\alpha = R(x_j, y)$, $R \in \mathsf{N}_\mathsf{R}^-$. The fact that $\mathcal{A}_{R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}}$ does not contain role assertions with individuals from $\mathsf{N}_\mathsf{I}(\mathcal{K})$ yields $R(\pi(x_j), \pi(y)) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W}$. Since $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$, for all assertions $R(\pi(x_j), b) \in \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W}$ with $b \in \mathsf{N}_\mathsf{I}(\Phi)$, contains the disjunct $(y = b)$ and elements from $\mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$ do not occur in this ABox, it also contains the disjunct $(y = \pi(y))$. Hence π is obviously as required.

- Let $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}} \cap \mathsf{N}_\mathsf{I}(\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}})$. That is, we regard elements from $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ of the form $a_{b\varrho}$, where $b \in \mathsf{N}_\mathsf{I}(\Phi)$—the elements from $\mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ that remain to be considered. We then must have $\pi(\alpha) \in \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$ since that kind of elements only occurs in this ABox. The arguments for showing that π satisfies $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$ are analogous to the ones applied for the case where $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{tree}}$ in the previous item. The only difference now is that $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$ can obviously contain role assertions with individuals from $\mathsf{N}_\mathsf{I}(\mathcal{K})$.

- For $\pi(x_j) \in \mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, we get $\pi(\alpha) \in \mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$, since the other ABoxes do not contain elements from $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$, and can assume a_j to be of the form $a_{[S]_\varrho}$. Observe that, then, no term is mapped to an element of $\mathsf{N}_\mathsf{I}(\Phi)$ since $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$ does not contain such elements (1). By the definition of $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$, $a_{[S]_\varrho}$ is (unambiguously) associated to the ABox $\mathcal{A}_{\exists S}$, one of the components of $\mathcal{A}^{\wp}_{R_{\mathsf{F}|\mathsf{o}}}$, which do not share elements of $\mathsf{N}_\mathsf{I}^{\mathsf{pro}}$. This yields $\pi(\alpha) \in \mathcal{A}_{\exists S}$ (2). The existence of the component $\mathcal{A}_{\exists S}$ further implies that there is an individual $a \in \mathsf{N}_\mathsf{I}(\mathcal{K}) \setminus \mathsf{N}_\mathsf{I}(\Phi)$ such that $\exists S(a) \in R_{\mathsf{F}|\mathsf{o}}$. By the definition of the latter set, there then is an index p such that $\langle \mathcal{O}, \mathcal{B}_{\mathsf{R}|\mathsf{o}} \cup \mathcal{A}_p \rangle \models \exists S(a)$. Lemma 6.27 thus yields that $\mathrm{TDB}(\mathfrak{A})$ satisfies $\mathsf{PRef}_{(\exists y.S(x,y),\mathcal{O}|\mathcal{B}_{\mathsf{R}|\mathsf{o}})}(p)$, which is a disjunct of $\mathsf{PRef}_{(\exists y.S(x,y),\mathcal{O}|\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(p)$. Note that, regarding role atoms, (1) implies that the case $t \in \mathsf{N}_\mathsf{I}(q)$ never applies.

  If α does not contain a variable that is quantified within q_k, then (2), $a \notin \mathsf{N}_\mathsf{I}(\Phi)$, and $\mathrm{TDB}(\mathfrak{A})$ satisfying $\mathsf{PRef}_{(\exists y.S(x,y),\mathcal{O}|\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(p)$ together yield that the replacement $\mathsf{rep}_{(\alpha|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$ is (trivially) satisfied under π.

If α contains a variable y that is quantified within q_k, we need to show $\pi(y) = [S]$, in addition. In this case, we have $\pi(y) \in \mathsf{N_I}(\mathcal{K})$ and that α is of the form $R(x_j, y)$, $R \in \mathsf{N_R^-}$. This means that the assertion $R(\pi(x_j), \pi(y))$ is contained in $\mathcal{A}^{\wp}_{R_{\mathsf{F|o}}}$ and, especially, in $\mathcal{A}_{\exists S(a)}$, by the definition of $\mathcal{A}^{\wp}_{R_{\mathsf{F|o}}}$ and the above observation that $\exists S(a) \in R_{\mathsf{F|o}}$. The (tree) shape of $\mathcal{A}_{\exists S(a)}$ and the definition of $\mathcal{A}_{\exists S}$, based on that ABox, then imply $R(\pi(x_j), [S]) \in \mathcal{A}_{\exists S}$ (and $\varrho = S$), $S \in \mathsf{N_R^-}$.

- We consider the case where the variable(s) and term(s) in α are not mapped to auxiliary elements of $\mathsf{N_I^{aux}} \cup \mathsf{N_I^{tree}} \cup \mathsf{N_I^{pro}}$, which means they are mapped to elements of $\mathsf{N_I}(\mathcal{K})$. If α is a flexible atom, then we have $\pi(\alpha) \in \mathcal{A}_{Q_W} \cup \mathcal{A}_i$ since all other ABoxes contain only rigid assertions. In case $\pi(\alpha) \in \mathcal{A}_{Q_W}$, we can apply arguments corresponding to (a subset of) those used for $\pi(x_j) \in \mathsf{N_I^{aux}}$ in the first item. The other case, $\pi(\alpha) \in \mathcal{A}_i$, is covered by Definitions 2.23 and 6.21. That is, the definition of TDB($\mathfrak{A}$) based on $\mathcal{A}_i$ yields that π is as required.

  If α is rigid, then we have $\pi(\alpha) \in \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_i$ because $\mathcal{A}_{R_\mathsf{F}}$ contains no assertion without auxiliary elements from $\mathsf{N_I^{tree}}$. We can also disregard $\mathcal{A}_{Q_W}$ since its rigid assertions are part of $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$, by definition, given that we assume $W \in \mathcal{W}$. Moreover, $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$ is addressed directly in the replacement; that part is especially complete since $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}}$ does not contain elements of $\mathsf{N_I}(\mathcal{K}) \setminus \mathsf{N_I}(\Phi)$. The other two ABoxes are covered by $\mathsf{PRef}_{(\alpha,\mathcal{O}|\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]})}(i)$, according to Lemma 6.28.

It can readily be checked that the above cases cover all kinds of elements π can map to. These observations show that π is a homomorphism of all our replacements into TDB($\mathfrak{A}$), and thus TDB($\mathfrak{A}$) $\models q_k$. Hence, TDB($\mathfrak{A}$) satisfies also our adaptation of $(q')^{\wp}$, which was to be shown.

($\Leftarrow$) We only sketch the proof for this direction since it is very similar to the above one. Since it neither differs for the two items to be proven, we again assume q' to be one of the two UCQs that are rewritten. By the semantics and the definition of the rewritings, we have a query q_k in the given rewriting that is an adaptation of a query $q^{\wp}$ in $(q')^{\wp}$ and satisfied in TDB($\mathfrak{A}$). We thus have a homomorphism π of q_k into TDB($\mathfrak{A}$) (i.e., w.r.t. the individual names and (existentially quantified) variables in q_k) and show that we can extend it adequately to cover the terms occurring in $q^{\wp}$. Because, q_k is a disjunction by construction, one of its disjuncts must be satisfied, by the semantics. We regard the individual names a_j from $\mathsf{N_I^{aux}} \cup \mathsf{N_I^{tree}} \cup \mathsf{N_I^{pro}}$ associated to the variables x_j in $q^{\wp}$ and to that disjunct. In particular, we extend π such that $\pi(x_j) = a_j$ for all these variables and subsequently show that this definition satisfies our purpose.

We ignore the "filtering" conjunct of $q^{\wp}$, for now, and consider an arbitrary conjunct representing the replacement for an atom $B(t)$ in $q^{\wp}$, in the disjunct under consideration, satisfied under π by assumption.

- If $\pi(t) \in \mathsf{N_I^{aux}} \cup \mathsf{N_I^{tree}}$, then *true* must be the replacement and $B(\pi(t))$ contained in $\mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_W} \cup \mathcal{A}_{R_{\mathsf{F|aux}[\mathcal{W}]}} \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$. By Definition 2.23, it can readily be checked that π then is as required.

- If $\pi(t) \in \mathsf{N_I^{pro}}$, then there are an $S \in \mathsf{N_R^-}(\mathcal{O}) \setminus \mathsf{N_{RR}}$ (i.e., ABoxes of the form $\mathcal{A}_{\exists S}$ are only defined for such roles), an individual $a \in \mathsf{N_I}(\mathcal{K}) \setminus \mathsf{N_I}(\Phi)$, and an index $i \in [0, n]$

such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_i \rangle \models \exists S(a)$ by Lemma 6.28. By Lemma 6.22, all rigid assertions about a that are consequences of that KB are contained in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$. By the definition of the latter, such assertions with a then are either contained in $\mathcal{B}_{\mathsf{R}|\mathsf{o}}$ or role assertions and consequences of a KB $\langle \mathcal{O}, \mathcal{A}_j \rangle$, $j \in [0, n]$; none of the other parts of $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ contains an element $a \notin \mathsf{N}_\mathsf{I}(\Phi)$, by their definitions. Since $\mathcal{B}_{\mathsf{R}|\mathsf{o}}$ includes the basic concept assertions corresponding to the former role assertions, by definition, and $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ cannot contain flexible assertions, we hence get $\langle \mathcal{O}, \mathcal{B}_{\mathsf{R}|\mathsf{o}} \cup \mathcal{A}_i \rangle \models \exists S(a)$ by Lemma 2.14. This means that $\exists S(a)$ is part of $R_{\mathsf{F}|\mathsf{o}}$, by definition, which yields $\mathcal{A}_{\exists S} \subseteq \mathcal{A}^{\wp}_{R_\mathsf{F}}$. We thus can again apply Definition 2.23, to show that π is as required.

- For the case where $\pi(t) \in \mathsf{N}_\mathsf{I}(\mathcal{K})$, Definition 2.23 can be applied regarding the first disjuncts, referring to $Q_{\mathsf{R}[\mathcal{W}]}$. Regarding the others, the definition of $\mathrm{TDB}(\mathfrak{A})$ and, respectively, Lemma 6.28 confirm the claim that the database satisfies $B(\pi(t))$.

The proof is similar for role atoms.

We lastly consider the "filtering" conjunct $\varphi_{\mathsf{filter}}$ of $q^\wp$, which is a conjunction of implications. By our extension of π in accordance with the individual names associated to the disjunct under consideration and the definitions of the predicate pro and the replacement of the corresponding atoms (i.e., those containing pro), we have that each atom occurring in $\varphi_{\mathsf{filter}}$ is satisfied in the database under the extended π iff its replacement is *true* and hence evaluates to true in $\mathrm{TDB}(\mathfrak{A})$ under π. Thus, our extension of π is as required. Again, the proof for the equality atoms is correspondingly. □

6.3.3 Rewriting r-Satisfiability

Based on the specific FO rewritings of KB satisfiability and UCQ entailment developed in the previous sections, we next define the FO formulas that capture r-satisfiability based on r-completeness.

For all $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$, $W \in \mathcal{W}$, and $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N}_\mathsf{I}(\Phi)\}$:

- $f_{(\mathrm{C1})}(i) := \neg q_{unsat(\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$;
- $f_{(\mathrm{C2})}(i) := \bigwedge_{p_j \in \overline{W}} \neg \mathsf{PRef}_{(\varphi_j,\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$;
- $f_{(\mathrm{C5})}(i) := \bigwedge_{\substack{\varphi \in Q^{\neg}_{\mathsf{R}},\\ \psi \text{ witness query}\\ \text{for } \varphi \text{ w.r.t. } \mathcal{O}}} \neg \mathsf{PRef}_{(\psi,\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}(i)$.

We integrate these formulas into the following abbreviation and subsequently describe how r-satisfiability can be tested.

$$\mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}(i) := f_{(\mathrm{C1})}(i) \wedge f_{(\mathrm{C2})}(i) \wedge f_{(\mathrm{C5})}(i).$$

Lemma 6.30 *For all $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ where $\mathcal{W} = \{W_1, \dots, W_k\}$, $\iota\colon [0, n] \to [1, k]$, and $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N}_\mathsf{I}(\Phi)\}$, the tuple*

$$(\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}, Q_{\mathsf{R}[\mathcal{W}]}, Q^{\neg}_{\mathsf{R}[\mathcal{W}]}, R_{\mathsf{F}[\mathcal{W},\mathcal{B}_\Phi]})$$

is r-complete w.r.t. $\mathcal{W}$ and ι iff the following hold:

- *For all* $i \in [0,n]$*, we have* $TDB(\mathfrak{A}) \models \mathsf{rSat}_{\mathcal{W},W_{\iota(i)},\mathcal{B}_\Phi}(i)$.
- *For all* $W \in \mathcal{W}$*, we have* $TDB(\mathfrak{A}) \models \mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}(-1)$.
- *For all* $S \in \mathsf{N}^-_\mathsf{R}(\mathcal{O}) \setminus \mathsf{N}^-_\mathsf{RR}$ *and* $a \in \mathsf{N}_\mathsf{I}(\Phi)$*, we have* $\exists S(a) \in \mathcal{B}_\Phi$ *iff there is an* $i \in [0,n]$ *such that* $TDB(\mathfrak{A}) \models \mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},W_{\iota(i)},\mathcal{B}_\Phi)}(i)$.

Proof. We consider Definition 6.8. The tuple generally satisfies some of the conditions by construction: $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$ is an ABox type; and $Q_{\mathsf{R}[\mathcal{W}]}$ and $Q^-_{\mathsf{R}[\mathcal{W}]}$ are in accordance with Conditions (C3) and (C4). Furthermore, Lemmas 6.29 and 2.24 show that the formulas in $\mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}(i)$ as considered in the first two items cover Conditions (C1), (C2), and (C5) adequately.

It remains to prove the equivalence between the satisfaction of Condition (C6) and the last item. For $R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$ and $R_{\mathsf{F}|\mathsf{o}}$, we have shown in the proof of Lemma 6.23 that they satisfy Condition (C6) by construction, independent of that item. We therefore focus on $R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$.

($\Leftarrow$) If the equivalence in the last item holds, then the definition of $R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ based on $\mathcal{B}_\Phi$ and Lemmas 6.29 and 2.24 yield that $\exists S(a) \in R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ iff there is an $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \cup \mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}} \rangle \models \exists S(a)$; note that $R_{\mathsf{F}|\mathsf{aux}[\mathcal{W}]}$ and $R_{\mathsf{F}|\mathsf{o}}$ do not contain relevant assertions. However, by Lemma 2.14, all parts of $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$ relevant to obtain such a conclusion are contained in $\mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]}$, given the definition of that ABox and that of $\mathcal{A}_{R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}}$. Note that the latter does not contain basic concept assertions on elements of $\mathsf{N}_\mathsf{I}(\Phi)$. We thus get that $\exists S(a) \in R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ iff there is an $i \in [0,n]$ such that $\langle \mathcal{O}, \mathcal{A}_{\mathsf{R}[\mathcal{W},\mathcal{B}_\Phi]} \cup \mathcal{A}_{Q_{\mathsf{R}[\mathcal{W}]}} \cup \mathcal{A}_{Q_{\iota(i)}} \cup \mathcal{A}_i \rangle \models \exists S(a)$, as required.

($\Rightarrow$) It is easy to see that, given the definition of $R_{\mathsf{F}|\Phi[\mathcal{B}_\Phi]}$ and Lemmas 6.29 and 2.24, Condition (C6) implies the condition in the last item. □

6.4 Data Complexity

In this section, we show that the low data complexity of query answering in *DL-Lite* does not increase dramatically in our temporal setting, for which we prove ALogTime-completeness. Though, FO rewritability thus is lost. As it is shown next, this holds already for the case without rigid names and regarding *DL-Lite*$_{core}$. The matching containment result is presented thereafter and based on several results from the previous sections.

6.4.1 Hardness

ALogTime-hardness can be shown by reducing the word problem of deterministic finite automata to TCQ entailment. The idea is to encode the given word into subsequent ABoxes and to emulate the state transitions in the TCQ, by requiring each to iterate to the next time point and to hold at all considered time points. Further, the TCQ considers the final state as a consequence of the latter, and the initial state is represented in the first ABox. In this way, the TCQ is entailed iff the automaton accepts the word. Note that we thus even do not need an ontology.

Theorem 6.31 *TCQ entailment in DL-Lite*$_{core}$ *is* ALogTime*-hard in data complexity, even if* $\mathsf{N}_\mathsf{RC} = \emptyset$ *and* $\mathsf{N}_\mathsf{RR} = \emptyset$.

Proof. It is well-known that every finite monoid M (i.e., a finite, closed set having an associative binary operation and an identity element) can be directly translated (in logarithmic time) to a deterministic finite automaton (DFA) that decides the word problem for that monoid, by regarding the elements of M as states and considering transitions according to the associative operation.[15] Moreover, for some such monoids (e.g., the group S5), this problem is complete for LogTime-uniform NC^1 under LogTime-uniform AC^0 reductions [BIS90, Cor. 10.2]; and LogTime-uniform NC^1 equals ALogTime [BIS90, Lem. 7.2].

We hence can establish ALogTime-hardness by considering an arbitrary DFA $\mathfrak{M}$ and reducing its word problem to TCQ entailment in logarithmic time. For that, we adapt a construction of [Art+15a, Thm. 9].

Let $\mathfrak{M}$ be a tuple of the form $(Q, \Sigma, \Delta, q_0, F)$, specifying the set of states Q, the alphabet Σ, the transition relation Δ, the initial state q_0, and the set of final states F. Regarding data complexity, the task is to specify a TCQ $\Phi_{\mathfrak{M}}$ based on $\mathfrak{M}$ and an ABox sequence $\mathfrak{A}_w$ based on an arbitrary input word $w \in \Sigma^*$ such that: $\mathfrak{M}$ accepts w iff $\langle \emptyset, \mathfrak{A}_w \rangle \models \Phi_{\mathfrak{M}}$. We consider concept names A_σ and Q_q for all characters σ of the input alphabet Σ and states $q \in Q$, respectively, and define the following TCQ:

$$\Phi_{\mathfrak{M}} := \Box_P \Big(\bigwedge_{q \to_\sigma q' \in \Delta} (Q_q(a) \wedge A_\sigma(a)) \to \bigcirc_F Q_{q'}(a) \Big) \to \bigvee_{q_f \in F} Q_{q_f}(a).$$

For a given input word $w = \sigma_0 \dots \sigma_{n-1}$, we then define the sequence $\mathfrak{A}_w = (\mathcal{A}_i)_{0 \leq i < n}$ of ABoxes as follows: $\mathcal{A}_0 := \{Q_{q_0}(a)\}$ and, for all $i \in [0, n]$, $\mathcal{A}_i := \{A_{\sigma_i}(a)\}$. It is easy to see that this reduction can be computed in logarithmic time.

Given that the semantics of TCQ entailment focus on time point n, it can readily be checked that the model of $\langle \emptyset, \mathfrak{A}_w \rangle$ that satisfies the premise of $\Phi_{\mathfrak{M}}$ at n represents the run of $\mathfrak{M}$ on w. Observe that there is only one such model relevant for entailment since $\mathfrak{M}$ is deterministic. Hence, $\mathfrak{M}$ accepts w iff all models of $\langle \emptyset, \mathfrak{A}_w \rangle$ that satisfy the premise also satisfy the disjunction $\bigvee_{q_f \in F} Q_{q_f}(a)$ at n. This is equivalent to the entailment $\langle \emptyset, \mathfrak{A}_w \rangle \models \Phi_{\mathfrak{M}}$. □

6.4.2 Containment

In this section, we finally provide an alternating Turing machine that solves our problem in logarithmic time based on Lemma 3.13. Since the latter considers satisfiability problems in both LTL and the DL, we first introduce some notation and establish last auxiliary results that facilitate our construction regarding the LTL part. The DL problems are solved by using the rewritability result from Section 6.3.

Separating LTL Satisfiability Testing

Similar to the algorithms proposed previously, our ATM is based on splitting the TCQ satisfiability testing into separate tests, each considering only a subset of time points (e.g., by regarding a set of formulas for each). Here, we again focus on Φ^{pa}; without loss of generality, we assume it to be separated. Specifically, we further separate the future

[15] We refer the reader to [BIS90] for details about monoids, groups, and the word problem in that context.

from the past formulas and then split the satisfiability testing of the latter. To this end, we again abstract from the given formula, similar to the idea of the propositional abstraction (see Definition 3.8).

Definition 6.32 (Propositional Boolean Abstraction) Let $\{q_1, \dots, q_o\}$ be a finite set of propositional variables such that there is a bijection $\cdot^{\text{ba}}$ mapping the top-level future and past subformulas occurring in Φ^{pa} to elements of that set.[16]

The *propositional Boolean abstraction* Φ^{ba} of Φ^{pa} w.r.t. $\cdot^{\text{ba}}$ is the propositional formula obtained from Φ^{pa} by replacing every top-level future and past subformula f in Φ^{pa} by f^{pa}. ♢

Additionally, we apply the following notation:

- We assume $\cdot^{\text{ba}}$ to be the propositional Boolean abstraction of Φ^{pa} w.r.t. the bijection $\cdot^{\text{ba}}$ mapping the top-level future and past subformulas $f_1, \dots, f_o$ contained in Φ^{pa} to propositions $q_1, \dots, q_o$ such that $q_i = f_i^{\text{ba}}$ for $i \in [1, o]$.
- $\mathcal{F}$ and $\mathcal{P}$ denote the sets of replaced top-level future and past subformulas in Φ^{pa}, respectively; that is, they form a partition of $\{f_1, \dots, f_o\}$.
- $\mathcal{V}$ denotes the set of all valuations $v \colon \{q_1, \dots, q_o\} \to \{\mathit{true}, \mathit{false}\}$ under which Φ^{ba} evaluates to *true*.
- We represent each $v \in \mathcal{V}$ also on the level of top-level formulas as follows:

$$\mathcal{P}^v := \{f_i \in \mathcal{P} \mid v(q_i) = \mathit{true}\} \cup \{\neg f_i \mid f_i \in \mathcal{P},\ v(q_i) = \mathit{false}\};$$

  $\mathcal{F}^v$ is defined analogously.
- For each $\mathcal{W} \subseteq 2^{\{p_1, \dots, p_m\}}$ and $v \in \mathcal{V}$, the set $\text{Fut}_{(\mathcal{W},v)}$ collects the worlds that may represent the beginning of an LTL model (restricted to $\mathcal{W}$) of the *future* subformulas $\mathcal{F}^v$ induced by v:

$$\text{Fut}_{(\mathcal{W},v)} := \{W \in \mathcal{W} \mid \exists \mathfrak{W} = (w_i)_{i \geq 0}, \forall i \geq 1 : w_i \in \mathcal{W}, \mathfrak{W} \models \bigwedge_{f \in \mathcal{F}^v} f, w_0 = W\}.$$

Note that all the above sets are independent of the data and can hence be regarded as constant. The below lemma describes the intention and establishes the correctness of our basic approach.

Lemma 6.33 *Let $\mathcal{W} = \{W_1, \dots, W_k\} \subseteq 2^{\{p_1, \dots, p_m\}}$ and $w_0, \dots, w_n \in \mathcal{W}$. The existence of an LTL structure $\mathfrak{W}$ that only contains worlds from $\mathcal{W}$, starts with $w_0, \dots, w_n$, and is such that $\mathfrak{W}, n \models \Phi^{\text{pa}}$ is equivalent to the existence of a valuation $v \in \mathcal{V}$ such that*

- $w_n \in \text{Fut}_{(\mathcal{W},v)}$ *and*
- $(w_0, \dots, w_n, w_n, \dots), n \models \bigwedge_{f \in \mathcal{P}^v} f$.

[16] Note that such a set and bijection obviously exist for the propositional abstraction of a TCQ.

Proof. ($\Rightarrow$) Given such an LTL structure $\mathfrak{W}$, the valuation v can be obtained by checking which elements of $\{f_1, \dots, f_o\}$ are satisfied at time point n, and the LTL structure needed for $\mathrm{Fut}_{(\mathcal{W},v)}$ is defined as the substructure of $\mathfrak{W}$ starting at n. Note that the satisfaction of the past formula $\bigwedge_{f \in \mathcal{P}^v} f$ in the LTL structure $(w_0, \dots, w_n, w_n, \dots)$ at time point n does not depend on any time point after n. We hence can choose arbitrary worlds from $\mathcal{W}$, such as w_n, for those time points.

($\Leftarrow$) It is easy to see that $\mathfrak{W}$ can be constructed by joining $w_0, \dots, w_n$ and the LTL structure obtained from the fact that $w_n \in \mathrm{Fut}_{(\mathcal{W},v)}$, since the satisfiability of past (future) subformulas at n is not affected by the worlds after (before) that time point.□

As outlined above, we further separate the satisfiability testing regarding the past subformulas. More precisely, we want to abstract similarly as for the future formulas and regard sets of worlds representing the satisfiability of a given set of past formulas. Since we however need to consider n specific such sets of worlds, we focus on *t-compatible types*, sets of past subformulas that are, respectively, satisfiable in one LTL structure at consecutive time points:

- A *type* for $\mathcal{P}$ is a subset T of $\mathrm{Clo}(\mathcal{P}) \cup \bigcup_{1 \leq i \leq m} \{p_i, \neg p_i\}$ that satisfies the following conditions:
  - for every $f \in \mathrm{Clo}(\mathcal{P})$, we have $f \in T$ iff $\neg f \notin T$;
  - for every $f \wedge g \in \mathrm{Clo}(\mathcal{P})$, we have $f \wedge g \in T$ iff $\{f, g\} \subseteq T$.

  Observe that we explicitly consider an extension of the closure since there may be propositions in $\{p_1, \dots, p_m\}$ that do not occur in $\mathcal{P}$, which are relevant when we regard time point n.
- The set $\mathrm{Typ}(\mathcal{P})$ represents the set of all types for $\mathcal{P}$.
- A type $T \in \mathrm{Typ}(\mathcal{P})$ is called *initial* if it does not contain formulas of the form $\bigcirc_P f$ and, for all $f \,\mathcal{S}\, g \in T$, we have $g \in T$.
- A pair $(T^{-1}, T) \in \mathrm{Typ}(\mathcal{P}) \times \mathrm{Typ}(\mathcal{P})$ is called *t-compatible* if the following hold:
  - $\bigcirc_P f \in T$ iff $f \in T^{-1}$,
  - $f \,\mathcal{S}\, g \in T$ iff either (i) $g \in T$, or (ii) $f \in T$ and $f \,\mathcal{S}\, g \in T^{-1}$.

Note that the decisions if a type is initial or if two types are t-compatible are independent of the data. Similarly, the above introduced sets can be regarded as constant. The next lemma shows that these new notions allow us to describe the satisfiability of the past subformulas at n by the existence of a specific mapping ι', which, based on types, distinguishes the worlds of an LTL structure. Note that this approach is in a certain way similar to the mapping ι regarded with TCQ satisfiability in Lemma 3.13.

Lemma 6.34 *For a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$, valuation $v \in \mathcal{V}$, and worlds $w_0, \dots, w_n \in \mathcal{W}$, we have*

$$(w_0, \dots, w_n, w_n, \dots), n \models \bigwedge_{f \in \mathcal{P}^v} f$$

iff there is a mapping $\iota' \colon [0, n] \to \mathrm{Typ}(\mathcal{P})$ as follows:

- *$\iota'(0)$ is initial and $\mathcal{P}^v \subseteq \iota'(n)$;*
- *for all $i \in [0, n[$, the pair $(\iota'(i), \iota'(i+1))$ is t-compatible;*
- *for all $i \in [0, n]$, we have $w_i \in \iota'(i) \cap \{p_1, \dots, p_m\}$.*

Proof. ($\Rightarrow$) The mapping ι' can be defined based on the first n worlds $w_0, \dots, w_n$ in the given structure, by considering exactly the past formulas satisfied in it at the respective time point. Note that other worlds do not have to be considered since the satisfiability of the past formulas between 0 and n does not depend on other time points, after n. For $i \in [0, n]$, this mapping is obviously compatible with the worlds w_i (formalized by the condition that $w_i \in \iota'(i) \cap \{p_1, \dots, p_m\}$), and it also satisfies the remaining conditions because of the temporal semantics.

($\Leftarrow$) Given ι', it can be shown by induction over the time points i, starting with $i = 0$, that, for all $f \in \mathrm{Clo}(\mathcal{P}) \cup \bigcup_{1 \leq i \leq m} \{p_i, \neg p_i\}$, we have $f \in \iota'(i)$ iff $(w_0, \dots, w_n, w_n, \dots), i \models f$. The condition on $\iota'(n)$ in the first item, which is assumed to be satisfied, then yields the claim. □

Observe that the conditions in the lemma are largely independent of the data (i.e., of n): Given a set $\mathcal{W} \subseteq 2^{\{p_1, \dots, p_m\}}$, which can be regarded as constant, it has basically to be ensured that there are t-compatible types for the time points between 0 and n and also corresponding worlds in $\mathcal{W}$. Most importantly, only two t-compatibility conditions have to be satisfied regarding each of these types, and each of these conditions influences only the selection of other types, in one direction of the time line—assuming the types from the first item in the lemma are selected correctly. This allows us to iteratively construct the mapping ι' in logarithmic time, by using an alternating Turing machine.

An Alternating Logarithmic-Time Turing Machine

We finally describe an alternating Turing machine that solves the TCQ satisfiability problem in *DL-Lite*$^{\mathcal{H}}_{horn}$ in logarithmic time in the size of the input, under the assumption that the ontology and the TCQ are fixed. Before describing the idea and going into the details of this particular machine, we briefly present some general specifics of TMs bounded by logarithmic time. The alternating version of such a machine is defined in the usual way, as an extension that discerns existential and universal states (see Definition 2.25).

Most importantly, the sublinear-time bound makes it necessary to provide a mechanism that allows the machine to reach all of the input in logarithmic time, which is not possible with the usual sequential scanning. We therefore adopt the random access model of [CKS81], where the symbols on the read-only input tape are accessed by writing the address of the symbol to be read (in binary) on a specific address tape, of which the TM then can access up to $\log l$ cells; l represents the length of the input, and the cells are assumed to hold the corresponding address. Next to those two tapes the machine may use a constant number of read/write work tapes.[17] To cover all the work

[17] Since Definition 2.25 is intended to introduce the concept of alternation on a higher level, we do not consider several tapes in it. According to [AB09, Claim 1.6], this extension leads however only to a quadratic increase in time.

tapes, the transition relation is then adapted correspondingly, and a *step* of the machine consists of reading one symbol from each tape, writing a symbol on each of the work tapes, moving each of the heads left or right one tape cell, or not at all, and entering a new state, in accordance with the transition relation.

We further apply the results of [BIS90] and assume that such TMs can do simple calculations as specified below.

Lemma 6.35 ([BIS90, Lem. 7.1]) *A deterministic log-time Turing Machine with input of length l can*

- *add and subtract numbers of $\mathcal{O}(\log l)$ bits,*
- *determine the logarithm of a binary number of $\mathcal{O}(\log l)$ bits, and*
- *subtract binary numbers of $\mathcal{O}(\log l)$ bits, compare such numbers, and compare them to 0.* □

The idea of our machine $\mathfrak{M}$ that decides the TCQ satisfiability problem integrates most of the results from the previous sections:

- The problem is equivalent to the existence of an r-satisfiable set $\mathcal{W} \in 2^{\{p_1,\dots,p_m\}}$ and mapping ι—focusing on propositions instead of CQs—such that Φ^{pa} is t-satisfiable w.r.t. them (Lemma 3.13).
- r-Satisfiability is FO rewritable if a set $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N_I}(\Phi)\}$ is considered additionally (Lemmas 6.30 and 6.9).
- t-Satisfiability can be decided based on a mapping $\tau' \colon [0,n] \to \mathrm{Typ}(\mathcal{P})$, which focuses on types instead of propositions (Lemmas 6.33 and 6.34), in a modular way.

The sets $\mathcal{W}$ and $\mathcal{B}_\Phi$ are guessed in the beginning. The mapping ι' is constructed while processing and such that it satisfies the conditions for t-satisfiability, and ι can be obtained from it. Based on ι, r-satisfiability is then checked in the last computation steps.

Note that most states we consider do not depend on the input data, which means that these computing steps do not influence the processing time significantly. On the other hand, $\log(n+1)$ specific states are crucial for our approach.

The mapping ι' is constructed in three phases. The two types $\iota'(0)$ and $\iota'(n)$ are existentially guessed initially. Then, $\mathfrak{M}$ continuously first guesses a t-compatible pair of types for the time points in the middle of the sequence for which ι' is to be defined (e.g., $0,\dots,n$ in the beginning) and, second, splits this sequence into half.[18] To ensure that ι' is fully constructed, this splitting happens in a universal state and, afterwards, a *left* and a *right copy* of $\mathfrak{M}$ are responsible for constructing ι' regarding the respective subsequences—excluding the time points at the borders of the sequences, which always have been guessed in previous steps. The copies proceed in this way until they

[18] For ease of presentation, we assume that $n+1$ is a power of 2. If this was not the case, the ATM would have to handle non-uniform divisions of the sequence $0,\dots,n$; this extension is possible but would complicate the presentation.

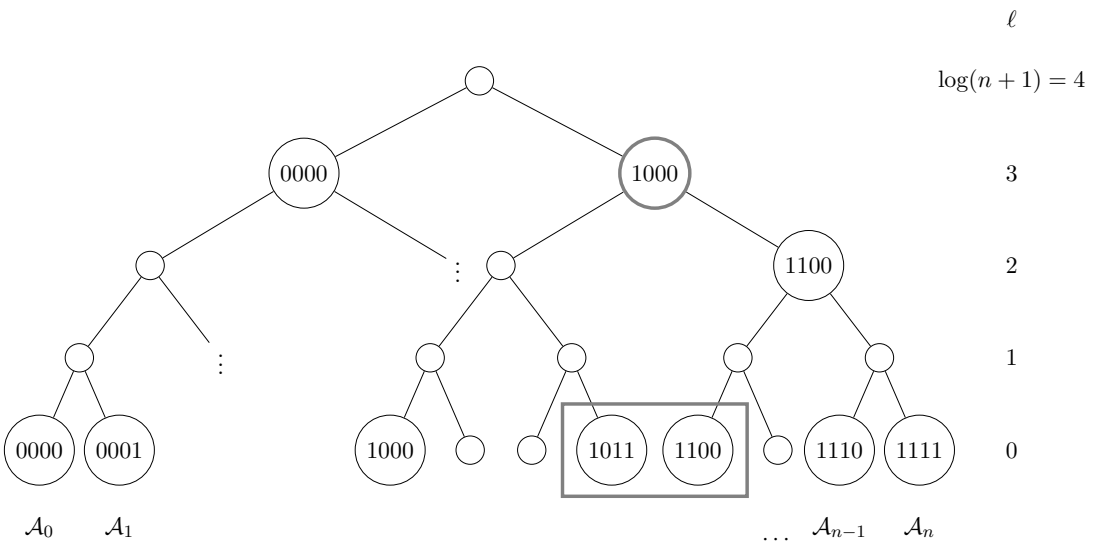

Figure 6.3: A sketch of the computation of the ATM for $n = 15$. The number ℓ denotes the current level of the computation tree. The nodes are labeled with the index i (in binary notation), which represents the computation path by designating the left border of the currently considered subsequence of $0, \dots, n$. The copy of the ATM designated by the marked node ($i = 1000$, $\ell = 3$) guesses a t-compatible pair $(\iota'(1011), \iota'(1100))$ of sets from $\mathrm{Typ}(\mathcal{P})$. The ATM then splits into two copies and each of those proceeds with one of the guessed types.

reach a sequence of two (consecutive) time points. For those, the types are then already given and, in particular, t-compatible regarding the respective other neighbor time points, w.r.t. the sequence $0, \dots, n$. It thus remains to ensure that the given pairs are t-compatible. Note that, in this construction, each value of ι' is guessed only once, which prevents conflicting guesses for one time point. Moreover, the copies do not have to know about the guessing that happens in other branches of the computation tree. We below give an example of the construction.

Example 6.36 Figure 6.3 illustrates the computation of $\mathfrak{M}$ given an ABox sequence with $n = 15$. We focus on the construction of ι'. Note that all circles apart from those in the bottom represent sequences of existential states followed by a universal state, in which the machine splits into two copies. The binary numbers on the labels are the starting (time) points of the respective sequences the copies are responsible for (i.e., regarding the construction of ι'). In an initial existential state $\iota'(0)$ and $\iota'(n)$ are guessed. After guessing $\iota'(\frac{n+1}{2} - 1)$ and $\iota'(\frac{n+1}{2})$, $\iota'(0111)$ and $\iota'(1000)$ in binary, $\mathfrak{M}$ splits into two copies that focus on the sequences $0, \dots, 7$ and $8, \dots, n$, respectively. The right copy thus regards $i = 1000$ as start, in binary, and guesses a t-compatible pair $(\iota'(i + 2^{\ell-1} - 1), \iota'(i + 2^{\ell-1}))$ of types, $\iota'(1011)$ and $\iota'(1100)$ in binary. Since all copies proceed in this way, those at level $\ell = 1$ focus on sequences of two time points for which the types have been guessed before. $\mathfrak{M}$ then accepts iff the corresponding pair of types

is t-compatible, which means that ι' is as required w.r.t. t-compability, and there are a set $\mathcal{W}$ and mapping ι such that $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$. $\diamondsuit$

After the last verification steps regarding ι' and two time points, each copy splits one last time so that every resulting copy can be associated to one time point $i \in [0, n]$. Together, these copies then construct the mapping ι by guessing a world that is in line with the type $\iota'(i)$, such that $w_i \in \mathcal{W} \cap \iota'(i) \cap \{p_1, \dots, p_m\}$.

Recall that testing r-satisfiability amounts to checking the satisfiability of FO formulas w.r.t. TDB($\mathfrak{A}$); and that the latter problem is in AC^0 [BIS90, Thm. 9.1], a subclass of LogTime. Hence, there are (deterministic) log-time TMs that decide the problems we consider in this regard, and $\mathfrak{M}$ can simulate these machines: for all $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$, $W \in \mathcal{W}$, $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N_I}(\Phi)\}$, $S \in \mathsf{N_R^-}(\mathcal{O}) \setminus \mathsf{N_{RR}^-}$, and $a \in \mathsf{N_I}(\Phi)$, we consider the machines $\mathfrak{M}_{\mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}}$ and $\mathfrak{M}_{\mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}}$. Observe that the overall complexity of our algorithm does not change because the number of considered satisfiability problems is constant and $\log(n) + \log(n) = 2\log(n)$. There are still two points that deserve special attention.

Regarding the second item in Lemma 6.30, $\mathfrak{M}$ has to consider satisfiability problems w.r.t. the empty ABox ($i = -1$) for the elements of $\mathcal{W}$, individually. To this end, it splits into $|\mathcal{W}|$ further copies that then verify the corresponding problems.

Regarding the last item in Lemma 6.30, we need to provide a solution to deal with the $n + 1$ satisfiability problems to really obtain a constant number of satisfiability problems to be considered. $\mathfrak{M}$ therefore guesses additionally, for each $\exists S(a) \in \mathcal{B}_\Phi$ with $S \in \mathsf{N_R} \setminus \mathsf{N_{RR}}$, at which time point i TDB($\mathfrak{A}$) satisfies $\mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},w_i,\mathcal{B}_\Phi)}(i)$. The elements of $\mathcal{B}_\Phi$ for which a copy of $\mathfrak{M}$ is responsible for are then propagated along the branches of the computation tree. Note that this guessing specifically does not consider all $n + 1$ time points at once but is done by dividing the set of assertions to be considered with each relevant split of $\mathfrak{M}$ (i.e., the splits represent the division of the sequence $0, \dots, n$).

Lastly, note that we assume the number n to be given with the input, written on the input tape in binary, at the beginning of the tape, and to be separated from the other input by a special marker symbol. This assumption is valid since n can be retrieved from the input via an FO query (e.g., a database could provide the number n in a view defined by the FO($<$) query $\neg\exists t.t > x$). The general input, the temporal database TDB($\mathfrak{A}$), is assumed to be given in the format required by the auxiliary machines that $\mathfrak{M}$ simulates (i.e., $\mathfrak{M}_{\mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}}$ with input i for deciding TDB($\mathfrak{A}$) $\models \mathsf{rSat}_{\mathcal{W},w_i,\mathcal{B}_\Phi}(i)$, etc.).

All information on Φ and $\mathcal{O}$ is fixed and encoded into the ATM itself. To this end $\mathfrak{M}$ uses corresponding states, transitions, and several work tapes, next to the input and address tape. Specifically, it comprises the tapes needed for the operations of Lemma 6.35; those required to simulate the machines $\mathfrak{M}_{\mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}}$ and $\mathfrak{M}_{\mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},W,\mathcal{B}_\Phi)}}$ for all $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$, $W \in \mathcal{W}$, $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N_I}(\Phi)\}$, $S \in \mathsf{N_R^-}(\mathcal{O}) \setminus \mathsf{N_{RR}^-}$, and $a \in \mathsf{N_I}(\Phi)$; and the below tapes. These tapes store the bounds of the sequence considered by the corresponding copy of $\mathfrak{M}$:

Tape 1 the index i of the left border of the considered subsequence of $0, \dots, n$, initially set to $i := -1$ and representing the branch of the computation tree;

Tape 2 the level ℓ in the computation tree.

The number of work tapes is thus constant. Tape 1 requires $\log(n+1)$ bits and Tape 2 requires $\mathcal{O}(\log\log(n+1))$ bits. For simplicity, we in the following use the term auxiliary machines to refer to all the machines $\mathfrak{M}$ simulates and also for the parts of $\mathfrak{M}$ that implement the operations described in Lemma 6.35.

We define the ATM as a tuple $\mathfrak{M} = (\mathcal{Q}, \Sigma, \Gamma, q_0, \Delta)$ (see also Definition 2.25):

- $\mathcal{Q}$ comprises all the states of the auxiliary machines and additional states that are specified below.
- Σ consists of symbols for representing n, the marker symbol used for separating n from the other input, and all symbols that may occur in the latter—its format depends on the requirements of the auxiliary machines.
- Γ comprises all symbols used by the auxiliary machines, especially those for implementing the counters i and ℓ.
- $q_0 \in \mathcal{Q}_\exists$.
- Regarding the transitions, first note that, with each transition, we always focus on one tape and assume the other tapes to remain the same (i.e., regarding such a tape, the transition replaces the symbol under the head by that symbol and does not move the head).

  Δ then contains the corresponding extensions of all transitions considered in the auxiliary machines and additional ones that we specify below, while describing the processing of the machine.

As outlined above, the processing of each copy of $\mathfrak{M}$ is based on sets $\mathcal{W}$ and $\mathcal{B}_\Phi$ guessed in the beginning. During the computation, additional information is characterizing the copies—a subset of $\mathcal{B}_\Phi$ and two types T_l and T_r. Furthermore, a valuation $v \in \mathcal{V}$ is associated with each copy, to guess the type $\iota'(n)$ correctly, such that $\mathcal{P}^v \subseteq \iota'(n)$. We implement this using corresponding states. Note that this is possible since the characterizing information itself is independent of the input. Specifically, we assume all states in $\mathcal{Q}$ apart from q_0 to be of the form q_ϑ, where ϑ denotes a (possibly indexed) tuple from a set Θ containing all tuples $(\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T_l, T_r)$ where:

- $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ such that $\mathcal{W} \neq \emptyset$;
- $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}),\ a \in \mathsf{N_I}(\Phi)\}$;
- $\mathcal{B}'_\Phi \subseteq \{\exists S(a) \in \mathcal{B}_\Phi \mid S \in \mathsf{N}^-_\mathsf{R}(\mathcal{O}) \setminus \mathsf{N}^-_\mathsf{RR}\}$;
- $v \in \mathcal{V}$;
- $T_l, T_r \in \mathrm{Typ}(\mathcal{P})$;

and similar tuples $(\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T)$ with $T \in \mathrm{Typ}(\mathcal{P})$. The latter are considered in the last computation steps, which focus on a single time point. Note that we assume that we also have corresponding versions of all states of the auxiliary machines; again, this assumption is possible because the information is independent of the input. In the following, we describe the processing of $\mathfrak{M}$ in detail.

- At start time, we have $i := -1$ and $\mathfrak{M}$ is in state $q_0 \in \mathcal{Q}_\exists$.
- $\mathfrak{M}$ then can go to all states $q_\vartheta \in \mathcal{Q}_\forall$ where $\vartheta = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T_l, T_r) \in \Theta$, T_l is initial, and $\mathcal{P}^v \subseteq T_r$.

  T_l and T_r represent $\iota'(0)$ and $\iota'(n)$, respectively.
- $\mathfrak{M}$ then must go to all states $q_{\vartheta'}$ as follows:
  - $\vartheta' = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T_l, T_r)_W$, $W \in \mathcal{W}$: $\mathfrak{M}$ uses the machine $\mathfrak{M}_{\mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}}$ with input i to check whether $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}(-1)$, rejects if this is not the case, and otherwise accepts (i.e., we assume that there are a corresponding existential and universal state, both without successor states).
  - ϑ' is of the same form as ϑ, but the corresponding state is existential; we denote it by $q^\exists_{\vartheta'} \in \mathcal{Q}_\exists$: $\mathfrak{M}$ proceeds as described in the following.
- $\mathfrak{M}$ increments the left index i and further initializes the counter ℓ such that $i := 0$ and $\ell := \log(n+1)$. We then assume $\mathfrak{M}$ to be in a state $q_\vartheta \in \mathcal{Q}_\exists$ as above.
- $\mathfrak{M}$ then continuously executes the below steps, while $\ell > 1$:
  - $\mathfrak{M}$ updates $\ell := \ell - 1$. We assume $\mathfrak{M}$ to be in a state $q_\vartheta \in \mathcal{Q}_\exists$ as above.
  - $\mathfrak{M}$ then can go to all states $q_{\vartheta'} \in \mathcal{Q}_\forall$ where

    $$\vartheta' = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T_l, T_r)_{\mathcal{B}^{(l)}_\Phi, \mathcal{B}^{(r)}_\Phi, T^{(l)}, T^{(r)}}$$

    $\mathcal{B}^{(l)}_\Phi$ and $\mathcal{B}^{(r)}_\Phi$ represent a partition of $\mathcal{B}'_\Phi$, and $(T^{(l)}, T^{(r)}) \in \mathrm{Typ}(\mathcal{P}) \times \mathrm{Typ}(\mathcal{P})$ is a t-compatible tuple.

    Note that, if there is no such t-compatible tuple, then q_ϑ is rejecting since it is existential.
  - From each such state $q_{\vartheta''}$, $\mathfrak{M}$ then must go to the two states $q_{\vartheta^{(l)}}, q_{\vartheta^{(r)}} \in \mathcal{Q}_\exists$ where

    $$\vartheta^{(l)} := (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}^{(l)}_\Phi, v, T_l, T^{(l)}),$$
    $$\vartheta^{(r)} := (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}^{(r)}_\Phi, v, T^{(r)}, T_r),$$

    and $\mathfrak{M}$ updates $i := i + 2^\ell$ in the latter state.

  In this way, we "store" the sets $T_l = \iota'(i)$ and $T_r = \iota'(i + 2^\ell - 1)$ in the state after each execution of the loop—as it was the case before the loop, after initialization.
- When ℓ has reached 1, $\mathfrak{M}$ proceeds similarly one final time:
  - $\mathfrak{M}$ updates $\ell := \ell - 1$. We then assume $\mathfrak{M}$ to be in a state $q_\vartheta \in \mathcal{Q}_\exists$ as above.
  - $\mathfrak{M}$ then can go to all states $q_{\vartheta'} \in \mathcal{Q}_\forall$ such that

    $$\vartheta' = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T_l, T_r)_{\mathcal{B}^{(l)}_\Phi, \mathcal{B}^{(r)}_\Phi},$$

    the tuple (T_l, T_r) is t-compatible, and $\mathcal{B}^{(l)}_\Phi$ and $\mathcal{B}^{(r)}_\Phi$ represent a partition of $\mathcal{B}'_\Phi$.

    If the tuple is not t-compatible, then q_ϑ is rejecting since it is existential.

– $\mathfrak{M}$ then must go to the states $q_{\vartheta^{(l)}}, q_{\vartheta^{(r)}} \in \mathcal{Q}_\exists$ where $\vartheta^{(l)} = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}^{(l)}_\Phi, v, T_l)$, $\vartheta^{(r)} = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}^{(r)}_\Phi, v, T_r)$, and $\mathfrak{M}$ updates $i := i + 1$ in the latter state. We assume $\mathfrak{M}$ to be in a state $q_\vartheta \in \mathcal{Q}_\exists$ where $\vartheta = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T)$.

Note that we now have that $\ell = 0$ and that T represents $\iota'(i)$.

- $\mathfrak{M}$ then can go to all states $q_{\vartheta'} \in \mathcal{Q}_\exists$ where $\vartheta' = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T)_{w_i}$ and we have $w_i \in T \cap \{p_1, \dots, p_m\}$.

  Again, q_ϑ is rejecting if such a world does not exist since it is existential.

- $\mathfrak{M}$ then can go to the state $q_{\vartheta''} := q_{\vartheta'}$ if $i \neq n$; if $i = n$ and $w_n \in \mathrm{Fut}_{(\mathcal{W},v)}$, $\mathfrak{M}$ can go to the state $q_{\vartheta''}$ where $\vartheta'' = (\mathcal{W}, \mathcal{B}_\Phi, \mathcal{B}'_\Phi, v, T)_{w_n}$; otherwise it rejects; in both cases we consider $q_{\vartheta''} \in \mathcal{Q}_\exists$.

- Lastly, $\mathfrak{M}$ checks if the below conditions are satisfied, by using the corresponding machines (i.e., $\mathfrak{M}_{\mathsf{rSat}_{\mathcal{W},w_i,\mathcal{B}_\Phi}}$ etc.) with i as additional input.
  – $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{rSat}_{\mathcal{W},w_i,\mathcal{B}_\Phi}(i)$.
  – For all $S \in \mathsf{N}^-_\mathsf{R}(\mathcal{O}) \setminus \mathsf{N}^-_\mathsf{RR}$ and $a \in \mathsf{N}_\mathsf{I}(\Phi)$ with $\exists S(a) \notin \mathcal{B}_\Phi$:
  $\mathrm{TDB}(\mathfrak{A}) \not\models \mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},w_i,\mathcal{B}_\Phi)}(i)$.
  – For all $\exists S(a) \in \mathcal{B}^{(i)}_\Phi$:
  $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},w_i,\mathcal{B}_\Phi)}(i)$.

  If any of these tests fails, $\mathfrak{M}$ rejects; otherwise it accepts.

A successful run confirms the satisfiability of Φ w.r.t. $\mathcal{K}$, as it is proven next.

Theorem 6.37 *TCQ entailment in DL-Lite$^{\mathcal{H}}_{horn}$ is in* ALogTime *in data complexity, even if* $\mathsf{N}_\mathsf{RR} \neq \emptyset$.

Proof. We regard the TCQ Φ, TKB $\mathcal{K} = \langle \mathcal{O}, \mathfrak{A} \rangle$, and temporal database $\mathrm{TDB}(\mathfrak{A})$ from above. Recall that $\mathrm{TDB}(\mathfrak{A})$ is simply a different representation of the ABox sequence $\mathfrak{A} = (\mathcal{A}_i)_{0 \leq i \leq n}$ and, particularly, of the same size.

Furthermore, it can readily be checked that $\mathfrak{M}$ terminates in logarithmic time in the size of the input:

- the number of states one copy of $\mathfrak{M}$ passes in one run of the loop is constant and the loop is run $\log(n+1) - 1$ times;
- the number of all other states explicitly mentioned above is also constant, and these states are passed at most once by a copy of $\mathfrak{M}$;
- all other states passed by a copy are due to calculations as in Lemma 6.35, based on existing log-time TMs.

We thus have that

the ATM $\mathfrak{M}$ accepts the input n and $\mathrm{TDB}(\mathfrak{A})$ (in logarithmic time) iff

there are sets $\mathcal{W} = \{W_1, \dots, W_k\} \subseteq 2^{\{p_1,\dots,p_m\}}$ and $\mathcal{B}_\Phi \subseteq \{B(a) \mid B \in \mathbb{B}(\mathcal{O}), a \in \mathsf{N}_\mathsf{I}(\Phi)\}$, a valuation $v \in \mathcal{V}$, a mapping $\iota' \colon [0, n] \to \mathrm{Typ}(\mathcal{P})$, and worlds $w_0, \dots, w_n \in \mathcal{W}$ as follows:

- for every $W \in \mathcal{W}$, we have $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{rSat}_{\mathcal{W},W,\mathcal{B}_\Phi}(-1)$;
- $\iota'(0)$ is initial and $\mathcal{P}^v \subseteq \iota'(n)$;
- for every $i \in [0,n]$, the pair $(\iota'(i), \iota'(i+1))$ is t-compatible;
- for every $i \in [0,n]$, we have $w_i \in \iota'(i) \cap \{p_1, \dots, p_m\}$;
- $w_n \in \mathrm{Fut}_{(\mathcal{W},v)}$;
- for every $i \in [0,n]$, we have $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{rSat}_{\mathcal{W},w_i,\mathcal{B}_\Phi}(i)$;
- for all $S \in \mathsf{N}_\mathsf{R}^-(\mathcal{O}) \setminus \mathsf{N}_\mathsf{RR}^-$ and $a \in \mathsf{N}_\mathsf{I}(\Phi)$, we have:
  $\exists S(a) \in \mathcal{B}_\Phi$ iff there is an $i \in [0,n]$ such that $\mathrm{TDB}(\mathfrak{A}) \models \mathsf{PRef}_{(\exists S(a),\mathcal{O}|\mathcal{W},w_i,\mathcal{B}_\Phi)}(i)$.

By Lemmas 6.34 and 6.33, this is equivalent to the existence of sets $\mathcal{W}$ and w_i as above and an LTL-structure $\mathfrak{W}$ as follows:

- $\mathfrak{W}$ only contains worlds from $\mathcal{W}$,
- $\mathfrak{W}$ starts with $w_0, \dots, w_n$,
- $\mathfrak{W}, n \models \Phi^{\mathrm{pa}}$.

Moreover, because of the condition that each w_i is an element of $\mathcal{W}$, the sequence $w_0, \dots, w_n$ can equivalently be expressed by a mapping $\iota\colon [0,n] \to [1,k]$ such that $w_i = W_{\iota(i)}$ for all $i \in [0,n]$.

The fact that the above is equivalent to the existence of sets $\mathcal{W}$ and ι such that Φ^{pa} is t-satisfiable w.r.t. $\mathcal{W}$ and ι, and $\mathcal{W}$ is r-satisfiable w.r.t. ι and $\mathcal{K}$ then follows from Definition 3.11 for t-satisfiability. Regarding r-satisfiability, we can apply Lemmas 6.23 and 6.30 describing the existence of an r-complete tuple w.r.t. $\mathcal{W}$ and ι and Lemma 6.9 linking this to r-satisfiability of $\mathcal{W}$. Finally, Lemma 3.13 yields the equivalence to the satisfiability of Φ w.r.t. $\mathcal{K}$. The claim thus follows from the fact that the class ALogTime is closed under complement (see [CKS81, Thm. 2.5]). □

We have thus completely classified TCQ entailment in several Horn fragments of *DL-Lite* regarding different settings with rigid symbols. In summary, we have shown that the main features of these logics, conjunction, inverse roles, role hierarchies, and unqualified existential restriction do not lead to "critical" interaction with LTL. This holds for both combined and data complexity. While the former stays the same, the latter only increases from in AC^0 to ALogTime-completeness.

7 Temporal Query Entailment in *DL-Lite*, Beyond the Horn Fragment

In this chapter, we show that for the Krom and Bool fragments of *DL-Lite*, the complexity results from Chapter 6, which focuses on the Horn fragment, do not apply any more, even if role hierarchies are disregarded. TCQ entailment actually gets as hard as for very expressive DLs such as $\mathcal{SHQ}$, and even harder.

We regard a Boolean TCQ Φ and a TKB $\mathcal{K} = \langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ written in a DL between *DL-Lite*$_{krom}$ and *DL-Lite*$_{bool}^{\mathcal{H}}$, depending on the context, and investigate the combined and data complexity in Sections 7.1 and 7.2, respectively. Throughout the chapter, we use the notation of Section 3.2.

7.1 Combined Complexity

For determining the combined complexity results, we first show a key observation: TCQ entailment in *DL-Lite*$_{bool}$ can be reduced to the problem in *DL-Lite*$_{krom}$. Specifically, the restricted disjunction allowed in *DL-Lite*$_{krom}$ together with TCQs allows to express several kinds of very expressive GCIs. The former then follows as a corollary and directly yields some hardness results based on atemporal query entailment for *DL-Lite*$_{bool}$. Subsequently, we investigate the combined complexity covering the different cases w.r.t. the rigid symbols.

As in the previous sections we often proceed according to Lemma 3.13 for obtaining containment results—for *DL-Lite*$_{krom}$. We therefore close this introduction with an auxiliary result on the satisfiability of conjunctions of CQ literals, which is important in this context.

Satisfiability of Conjunctions of CQ literals

We show that, for a Boolean TCQ that is a conjunction of CQ literals, the satisfiability w.r.t. a classical KB can be decided in ExpTime w.r.t. combined complexity. This can be proven using a reduction to UCQ non-entailment by instantiating the positive literals, as we did it for $\mathcal{EL}$ (see the proof of Lemma 5.1). This approach is independent of the DL and proposed in [BBL15b] (see the proof of Theorem 4.1 in that paper).

Lemma 7.1 *For a Boolean conjunction Ψ of CQ literals and a (atemporal) knowledge base $\mathcal{K} = \langle \mathcal{O}, \mathcal{A} \rangle$ in DL-Lite*$_{krom}$*, the satisfiability of Ψ w.r.t. $\mathcal{K}$ can be decided in* ExpTime *w.r.t. combined complexity.*

Proof. As outlined above, the idea is to reduce the problem to UCQ non-entailment by instantiating the positive literals, as it is described in the proof of Lemma 5.1. The ExpTime complexity for *DL-Lite*$_{krom}$ then follows from the fact that UCQ non-entailment w.r.t. a set of facts and so-called frontier-one disjunctive inclusion depen-

GCI	TCQ
$\exists R.A_1 \sqsubseteq A_2$	$\neg\exists x, y.R(x,y) \wedge A_1(y) \wedge \overline{A}_2(x)$
$A_1 \sqsubseteq \forall R.A_2$	$\neg\exists x, y.A_1(x) \wedge R(x,y) \wedge \overline{A}_2(y)$
$A_1 \sqcap \cdots \sqcap A_m \sqsubseteq A_{m+1} \sqcup \cdots \sqcup A_{m+n}$	$\neg\exists x.A_1(x) \wedge \cdots \wedge A_m(x) \wedge \overline{A}_{m+1}(x) \wedge \ldots \overline{A}_{m+n}(x)$

Figure 7.1: The representation of complex GCIs in *DL-Lite*$_{krom}$ by TCQs.

dencies can be decided in ExpTime [BMP13, Thm. 8]. This kind of dependencies are first-order formulas of the form $\forall\vec{x}.\varphi(\vec{x}) \rightarrow \bigvee_{i=1}^{\ell} \exists\vec{y}.\psi_i(\vec{z},\vec{y_i})$, where φ is a conjunction of binary atoms, all ψ_i with $i \in [1,\ell]$ are binary atoms, $\vec{x}$ and all $\vec{y_i}$ where $i \in [1,\ell]$ are pairwise disjoint sets of variables of cardinality two, and $\vec{z} \subseteq \vec{x}$ has maximally a cardinality of one. Observe that CIs of the form $A \sqcap B \sqsubseteq \bot$ cannot be expressed with such dependencies. However, considering $\mathcal{A}'$ to be an ABox containing instantiations of the positive CQ literals and Ψ' to be the disjunction of the negative ones, the entailment $\langle\mathcal{O}, \mathcal{A} \cup \mathcal{A}'\rangle \models \Psi'$ is equivalent to $\langle\mathcal{O}', \mathcal{A} \cup \mathcal{A}'\rangle \models \Psi' \vee \Psi_\bot$, where $\mathcal{O}'$ is the maximal subset of $\mathcal{O}$ that does not contain the CIs of the form $A \sqcap B \sqsubseteq \bot$, and $\Psi_\bot$ is a disjunction containing a CQ $\exists x.A(x) \wedge B(x)$, for each of these CIs. It is easy to see that the combined complexity of in ExpTime is retained by this reformulation. □

7.1.1 Reducing *DL-Lite*$_{bool}$ to *DL-Lite*$_{krom}$ Entailment

We show that CIs with disjunction that are of the form $\top \sqsubseteq A \sqcup \overline{A}$, which are allowed in *DL-Lite*$_{krom}$, together with TCQs can express several kinds of GCIs that are not allowed in *DL-Lite*$_{krom}$ and especially cover *DL-Lite*$_{bool}$ (see Figure 7.1). More precisely, we use TCQs that are negated CQs to rule out the satisfaction of combinations of concepts if it would contradict such a GCI. To describe these combinations, we use (fresh) symbols of the form $\overline{A}$, representing the complements of given concept names A; that is, CIs as the above one and corresponding disjointness axioms are added to the ontology. The following lemma specifies our approach.

Lemma 7.2 *Let $(C \sqsubseteq D, \neg\varphi)$ be one of the pairs of a GCI and a TCQ given in Figure 7.1, and let $\mathcal{I}$ be a model of $\top \sqsubseteq A_i \sqcup \overline{A}_i$ and $A_i \sqcap \overline{A}_i \sqsubseteq \bot$ for all concept names A_i occurring in D. Then, we have that $\mathcal{I} \models C \sqsubseteq D$ iff $\mathcal{I} \models \neg\varphi$.*

Proof. ($\Rightarrow$) We assume $\mathcal{I} \not\models \neg\varphi$, which yields $\mathcal{I} \models \varphi$, and hence that there is a corresponding homomorphism by Definition 3.5. Especially, observe that the atoms in the CQ φ always refer to the concepts and roles of the corresponding GCI $C \sqsubseteq D$ in the same way, so that C and $\neg D$ are modeled in the CQ. Thus, the shape of φ together with our assumption that $\mathcal{I}$ satisfies the CIs w.r.t. $\overline{D}$ and the semantics of the constructor $\forall$[1] yield that there is an element e in the domain of $\mathcal{I}$ such that $e \in C^\mathcal{I}$ and $e \notin D^\mathcal{I}$. This directly yields $C^\mathcal{I} \not\subseteq D^\mathcal{I}$, and thus $\mathcal{I} \not\models C \sqsubseteq D$. ($\Leftarrow$) The proof for this direction is by dual arguments. □

We thus can use GCIs as described above in the ontology we construct for proving hardness of TCQ entailment. More precisely, by the above lemma, we have:

$$\langle\mathcal{O}, (\mathcal{A}_i)_{0\leq i\leq n}\rangle \models \Phi \text{ iff } \langle\mathcal{O}', (\mathcal{A}_i)_{0\leq i\leq n}\rangle \models ((\Box_P \Box_F \Psi) \rightarrow \Phi), \text{ where}$$

[1] In an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, a concept $\forall R.C$ is interpreted as $\{x \in \Delta^\mathcal{I} \mid \forall(x,y) \in R^\mathcal{I} : y \in C^\mathcal{I}\}$.

- $\mathcal{O}'$ is obtained from $\mathcal{O}$ by removing all GCIs of the forms listed in Figure 7.1 and adding the CIs to express the corresponding complement concepts, and
- Ψ is the conjunction of the negated CQs simulating the removed GCIs.

With the same construction, Φ is satisfiable w.r.t. $\langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$ iff $(\Box_P \Box_F \Psi) \wedge \Phi$ is satisfiable w.r.t. $\langle \mathcal{O}', (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$. This means that we can also use CIs of *DL-Lite*$_{bool}$ in *DL-Lite*$_{krom}$ for our purpose, which yields the following corollary.

Corollary 7.3 *TCQ entailment in DL-Lite*$_{bool}$ *can be polynomially reduced to TCQ entailment in DL-Lite*$_{krom}$*.*

This result implies that TCQ entailment in *DL-Lite*$_{bool}$ can be decided by applying algorithms solving the problem for *DL-Lite*$_{krom}$ and that the complexity does not increase. On the other hand, we can apply the kinds of GCIs from Figure 7.1 when proving hardness results for TCQ entailment in *DL-Lite*$_{krom}$. Additionally, observe that more complex GCIs with nested conjunctions and disjunctions can be reduced to those forms by introducing appropriate abbreviations, as long as qualified existential restriction (i.e., in concepts of the form $\exists R.A$) only appears on the left and qualified value restriction (i.e., in concepts of the form $\forall R.A$) only on the right-hand side.[2] This is demonstrated in the following example; note that the transformation therein is polynomial.

Example 7.4 The GCI

$$A_1 \sqcup A_2 \sqcup \exists R_1.A_3 \sqsubseteq A_4 \sqcup \forall R_1.(A_1 \sqcap \exists R_2)$$

can be expressed by the following GCIs, assuming $A'_5, \dots, A'_7$ to be fresh concept names:

$$A_1 \sqsubseteq A_4 \sqcup A'_5,\ A_2 \sqsubseteq A_4 \sqcup A'_5,\ A'_6 \sqsubseteq A_4 \sqcup A'_5,$$
$$\exists R_1.A_3 \sqsubseteq A'_6,\ A'_5 \sqsubseteq \forall R_1.A'_7,\ A'_7 \sqsubseteq A_1,\ A'_7 \sqsubseteq \exists R_2.$$

These GCIs can then, in turn, be simulated by negated CQs as described in Figure 7.1.$\diamondsuit$

We hence can also apply such complex GCIs in the following proofs; with all these applications, we will outline the corresponding transformation.

7.1.2 Without Rigid Names

Given the results from the previous section (see Corollary 7.3), we directly get two rather strong hardness results from atemporal query answering, even *without considering rigid symbols*. More precisely, the ExpTime-hardness of UCQ entailment in *DL-Lite*$_{bool}$ [Bou+16, Thm. 8.2] and the 2-ExpTime-hardness of UCQ entailment in *DL-Lite*$^{\mathcal{H}}_{bool}$ [Bou+16, Thm. 8.1] yield corresponding hardness results for TCQ entailment in *DL-Lite*$_{krom}$ and *DL-Lite*$^{\mathcal{H}}_{krom}$, respectively.

Corollary 7.5 *Regarding combined complexity, TCQ entailment is*

- ExpTime*-hard in DL-Lite*$_{krom}$ *and*

[2] Recall that qualified existential restriction on the right-hand side can be expressed in *DL-Lite* if role inclusions are allowed (see the beginning of Chapter 6). But we disregard those here.

- 2-ExpTime-*hard in* $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{krom}$,

even if $\mathsf{N_{RC}} = \emptyset$ *and* $\mathsf{N_{RR}} = \emptyset$.

Note that TCQ entailment in $\mathcal{SHIQ}$ can be decided in 2-ExpTime, including rigid symbols [BBL15a, Thm. 12]. Since $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{bool}$ is a subset of this DL, we can focus on $\mathit{DL\text{-}Lite}_{krom}$ and $\mathit{DL\text{-}Lite}_{bool}$ in the remaining considerations w.r.t. combined complexity.

Regarding the case without rigid symbols, Lemma 3.13 yields that TCQ satisfiability can be decided in exponential time, given the above auxiliary results. Hence, the entailment problem can be solved similarly.

Theorem 7.6 *TCQ entailment in* $\mathit{DL\text{-}Lite}_{bool}$ *is in* ExpTime *in combined complexity if* $\mathsf{N_{RC}} = \emptyset$ *and* $\mathsf{N_{RR}} = \emptyset$.

Proof. By Corollary 7.3, it suffices to describe a decision procedure for $\mathit{DL\text{-}Lite}_{krom}$, which can be done by following Lemma 3.13. Assuming Φ to be satisfiable w.r.t. $\langle \mathcal{O}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$, we in the following show that we can construct a set $\mathcal{W} \subseteq 2^{\{p_1, \dots, p_m\}}$ as required by the lemma. By Lemma 3.7, we can restrict the focus to the TKB $\langle \mathcal{O}, \emptyset \rangle$ and hence disregard ι by Fact 3.15. To check the r-satisfiability of a set $\mathcal{W} = \{W_1, \dots, W_k\}$ as proposed in Lemma 3.14, it suffices to check satisfiability of the conjunctions $\chi_i^{(i)}$ with $i \in [1, k]$ individually by Lemma 3.16—without rigid names, it is impossible to enforce any dependency between the elements of $\mathcal{W}$. Hence, it suffices to define $\mathcal{W}$ as the set of *all* those of these sets for which χ_i is satisfiable w.r.t. $\mathcal{O}$. According to Lemma 7.1, this can be done in exponential time.

The deterministic definition of $\mathcal{W}$ as the maximal possible set satisfying the conditions implies that Lemma 3.13 applies by Fact 3.15, which means that the procedure is sound and complete. We thus can decide TCQ satisfiability in deterministic exponential time in the size of all the input. The same then holds for entailment. □

7.1.3 With Rigid Concept Names

Regarding the case with rigid concept names but without rigid role names, we show that TCQ satisfiability is NExpTime-hard. In fact, this is a direct consequence of the proof of Theorem 4.8, where NExpTime-hardness is shown for the satisfiability problem of formulas in $\mathit{DL\text{-}Lite}_{horn}$-LTL, similarly regarding rigid concept names. In a nutshell, the proof combines the nondeterminism and exponentiality expressible in LTL with the DL features: the rigid symbols may "save" the nondeterministic choices invariant to time and CIs allow to express constraints on them.

The proof of Theorem 4.8 is a reduction from a NExpTime-hard variant of the domino problem, constructing a corresponding $\mathit{DL\text{-}Lite}_{horn}$-LTL formula $\Phi_{\mathcal{D},I}$. Given the following two observations, $\Phi_{\mathcal{D},I}$ can be regarded as a TCQ and, taking a $\mathit{DL\text{-}Lite}_{krom}$ ontology $\mathcal{O}$ as specified below into account, the result can be directly applied here:

- The assertions in the constructed $\mathit{DL\text{-}Lite}_{horn}$-LTL formula $\Phi_{\mathcal{D},I}$ are of the form $A(a)$ for $A \in \mathsf{N_C}$ and $a \in \mathsf{N_I}$ and hence can be regarded as CQs.
- The CIs occurring in $\Phi_{\mathcal{D},I}$ are of the form $\top \sqsubseteq A_1$, $A_1 \sqsubseteq \bot$, or $A_1 \sqcap \dots \sqcap A_\ell \sqsubseteq A_{\ell+1}$, $A_1, \dots, A_{\ell+1} \in \mathsf{N_C}$, and thus Lemma 7.2 yields that they can be replaced by

negated CQs according to Figure 7.1 without affecting the semantics (see also the part above Corollary 7.3). $\mathcal{O}$ then collects the CIs that constrain the interpretation of the auxiliary names w.r.t. their complements.

Theorem 7.7 *TCQ entailment in DL-Lite$_{krom}$ is* co-NExpTime*-hard w.r.t. combined complexity if* $\mathsf{N_{RC}} \neq \emptyset$*, even if* $\mathsf{N_{RR}} = \emptyset$*.*

For proving containment in co-NExpTime regarding *DL-Lite$_{bool}$*, we apply Lemma 3.13 and the above auxiliary results again. The crucial point for investigating TCQ satisfiability regarding the lemma is the satisfiability testing of the conjunctions of CQ literals, looking for models that agree on the interpretation of rigid concepts. But we can guess all combinations of rigid concept names that are instantiated in some of these models together with one such combination, for each named individual in the TKB, in exponential time. If we include this information, we can separate the satisfiability testing of the different conjunctions, according to [BBL15b, Lem. 6.2].

Theorem 7.8 *TCQ entailment in DL-Lite$_{bool}$ is in* co-NExpTime *regarding combined complexity if* $\mathsf{N_{RR}} = \emptyset$*, even if* $\mathsf{N_{RC}} \neq \emptyset$*.*

Proof. We regard the satisfiability of Φ w.r.t. $\mathcal{K}$, focus on *DL-Lite$_{krom}$* (see Corollary 7.3), and argument based on Lemma 3.13.

- As in the proof of Theorem 7.6, we can assume $\mathcal{K}$ to be of the form $\langle \mathcal{O}, \emptyset \rangle$, since integrating the ABoxes into the TCQ does not influence combined complexity. By Lemma 3.7, we can assume $\mathcal{K}$ to be of the form $\langle \mathcal{O}, \emptyset \rangle$, since the integration of the ABoxes into the TCQ does not influence combined complexity. This means that the selection of an appropriate mapping ι is trivial because $\iota(0)$ can be chosen arbitrarily and hence such that $W_{\iota(0)}$ is contained in the set $\mathcal{W}$ we construct below; recall that $\mathcal{W}$ must not be empty (see also Fact 3.15).

- We can obviously guess a set $\mathcal{W} \subseteq 2^{\{p_1,\dots,p_m\}}$ in exponential time, and the same holds for checking t-satisfiability of Φ^{pa} w.r.t. this set and a corresponding mapping ι [BBL15b, Lem. 4.12].

- For checking r-satisfiability, we guess a set $\mathcal{T} \subseteq 2^{\mathsf{N_{RC}}(\mathcal{O})}$, which specifies the combinations of rigid names that are allowed to be satisfied by domain elements in the models of the considered conjunctions, and a mapping $\tau\colon \mathsf{N_I}(\Phi) \to \mathcal{T}$ that fixes the rigid concepts each individual occurring in $\mathcal{K}$ instantiates—note that there are similarities to the ABox types we considered in the previous sections. Based on τ, we define a polynomially-sized ontology $\mathcal{O}_\tau$ and CQ ψ_τ, of exponential size, as follows:

$$\mathcal{O}_\tau := \{A_{\tau(a)} \equiv C_{\tau(a)} \mid a \in \mathsf{N_I}(\Phi)\} \cup \bigcup_{A \in \mathsf{N_{RC}}(\mathcal{O})} \{\top \sqsubseteq A \sqcup \overline{A},\ A \sqcap \overline{A} \sqsubseteq \bot\},$$

$$\psi_\tau := \bigwedge_{a \in \mathsf{N_I}(\Phi)} A_{\tau(a)}(a)$$

where $\equiv$ is an abbreviation for both $\sqsubseteq$ and $\sqsupseteq$, and C_T with $T \subseteq \mathsf{N_{RC}}(\mathcal{O})$ is defined as $C_T := \bigsqcap_{A \in T} A \sqcap \bigsqcap_{A \in \mathsf{N_{RC}}(\mathcal{O}) \setminus T} \overline{A}$. We further say that an interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ *respects* $\mathcal{T}$ if

$$\mathcal{T} = \{T \subseteq \mathsf{N_{RC}}(\mathcal{O}) \mid \exists e \in (C_T)^{\mathcal{J}}\}.$$

In [BBL15b, Lem. 6.2], it is shown that $\mathcal{W}$ is r-satisfiable w.r.t. $\mathcal{K}$ iff there are a set $\mathcal{T}$ and mapping τ as above such that each conjunction $\chi_i \wedge \psi_\tau$ with $i \in [1, k]$ has a model w.r.t. $\mathcal{O} \cup \mathcal{O}_\tau$ that respects $\mathcal{T}$. The proof considers the DL $\mathcal{SHQ}$, but similarly holds regarding *DL-Lite*$_{krom}$.

Although it seems that the claimed NExpTime result then directly follows from Lemma 7.1 stating that conjunctions of CQ literals can be decided in exponential time, this is not the case. The inclusion of $\mathcal{T}$ causes an exponential blowup (i.e., the fact that it is to be respected). For that reason, we consider the proof of [BMP13, Thm. 8], which addresses UCQ entailment, in more detail; recall that we refer to that theorem in the proof of Lemma 7.1. In that paper, an exponentially large looping tree automaton is constructed that recognizes exactly those (forest-shaped) canonical models of the KB—in a wider sense—that do not satisfy the given UCQ. We integrate the check that the interpretations respect $\mathcal{T}$ into the automaton. To this end, we adapt the automaton to accept arbitrary models. Still, the restriction to tree-shaped models is without loss of generality. Then, we restrict the state set to consider only models where every domain element satisfies some C_T with $T \in \mathcal{T}$. To ensure that each $T \in \mathcal{T}$ is represented somewhere in the model, we specifically check $|\mathcal{T}|$ variants of this automaton for emptiness, each of them includes an ABox of the form $\{A(a) \mid A \in T\} \cup \{\overline{A}(a) \mid A \in \mathsf{N_{RC}}(\mathcal{O}) \setminus T\}$, where a is a fresh individual name, and is of polynomial size. The disjoint union of all resulting interpretations is still a model of the original KB that does not satisfy the UCQ (i.e., if none of the variants satisfies the emptiness check, then there is such an interpretation for each of them). It can readily be checked that this modified procedure for deciding UCQ non-entailment is sound and complete given the result of [BMP13, Thm. 8]. The satisfiability of the conjunctions thus can be decided in exponential time, because the constructed automata are of exponential size and emptiness of looping tree automata can be decided in polynomial time [VW86, Thm. 2.2].

The co-NExpTime result for TCQ entailment then directly follows from the above considerations. □

7.1.4 With Rigid Role Names

The previous section has shown that the LTL features for discerning exponentially many time points and nondeterministically selecting axioms at each of them lead to NExpTime-hardness if the DL part provides nondeterminism as allowed in *DL-Lite*$_{krom}$ and rigid concept names; the latter allow to correspondingly discern exponentially many concepts and hence kinds of individuals that can be addressed invariant to time. In this section, we show how rigid roles may augment this interaction and add an exponential factor to the complexity. The point is that the LTL allows to loop over the exponentially

many time points—this is not due to the new features—and the rigid roles allow to relate exponentially many corresponding individuals invariant to time. This yields an infinite chain whose sequence of individuals mirrors the repeating time points, so that exponentially many subsequent individuals differ in the kind. Since individuals of one kind can be addressed in the ontology at a distinct time point in the loop, they can influence all their successors in the chain independently of individuals of a different kind—also, regarding rigid symbols. In what follows, we show that this interaction leads to 2-ExpTime-hardness of TCQ satisfiability. Recall that containment in 2-ExpTime follows from the corresponding result for the DL $\mathcal{SHIQ}$ [BBL15a, Thm. 12].

We prove 2-ExpTime-hardness of TCQ satisfiability and hence entailment by reducing the word problem of exponentially space-bounded alternating Turing machines. The idea is based on the features outlined above. More precisely, a chain as described may represent a computation of the machine, and the exponentially long consecutive sequences of individuals of different kind represent its configurations. The nondeterminism provided in *DL-Lite*$_{krom}$ allows to require additional rigid relations according to the transition relation and hence to model the entire computation tree.

Note that the latter is not possible in, for example, $\mathcal{EL}$-LTL, *DL-Lite*$_{bool}$-LTL, or with TCQs in $\mathcal{EL}$, for which we similarly have NExpTime-hardness, but also completeness.

Theorem 7.9 *TCQ entailment in DL-Lite*$_{krom}$ *is* 2-ExpTime*-hard w.r.t. combined complexity if* $\mathsf{N_{RR}} \neq \emptyset$.

Proof. We adapt a reduction proposed in [BGL12] (see the proof of Theorem 4.1), where the word problem for exponentially space-bounded alternating Turing machines is reduced to the satisfiability problem in $\mathcal{ALC}$-LTL with global GCIs and rigid names. While the assertions in the proposed $\mathcal{ALC}$-LTL formula can be directly regarded as conjuncts of a TCQ, the global GCIs cannot all be transferred into a *DL-Lite*$_{krom}$ ontology since $\mathcal{ALC}$ is much more expressive than *DL-Lite*$_{krom}$. However, we show how some of the critical GCIs can be adapted to comply with the shapes given in Figure 7.1, and how the remaining ones—with qualified existential restrictions on the right-hand-side—can be replaced by equivalent new constructions. Note that the latter are inspired by [KRH13] (see Section 6.2 in that paper).

As in the original proof, we assume w.l.o.g. that an ATM never moves to the left when it is on the left-most tape cell; that there are an accepting state q_a and a rejecting state q_r, designating accepting and rejecting configurations, respectively; that any configuration where the state is neither q_a nor q_r has at least one successor configuration; and that all computations of an ATM are finite (see [CKS81, Thm. 2.6]). We disregard transitions that do not move the head (N). Further, we may assume that the length of every computation on a word $w \in \Sigma^k$ is bounded by 2^{2^k}, and that every configuration in such a computation can be represented using $\leq 2^k$ symbols, plus one to represent the state.

According to [CKS81, Cor. 3.5], there is an exponentially space-bounded alternating TM $\mathfrak{M} = (\mathcal{Q}, \Sigma, \Gamma, q_0, \Delta)$ with only finite computations for which the word problem is 2-ExpTime-hard. We show that this problem can be reduced to TCQ satisfiability in *DL-Lite*$_{krom}$ with rigid role names.

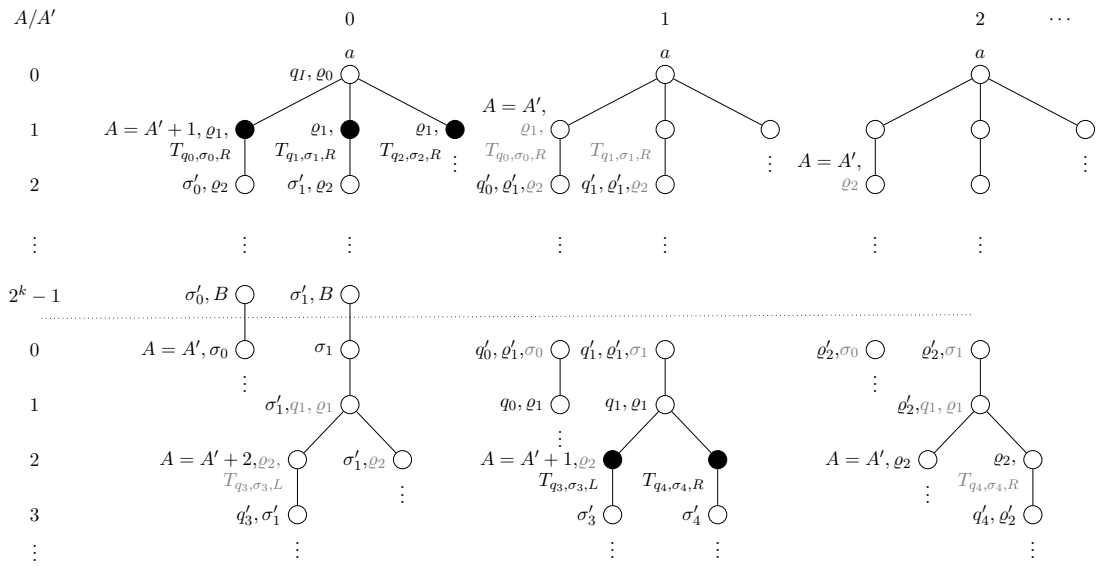

Figure 7.2: A sketch of the modeling of an exemplary computation tree of the ATM. The tree nodes represent domain individuals and are labeled with relevant concepts. The named individual a represents the first cell in the initial configuration, q_I is the initial machine state, and ϱ_0 the symbol in the first tape cell. Some of the rigid concepts are printed in gray to differentiate those time points from the time points where they are inferred. The figure also abstracts from the fact that the temporal counter A' has to be considered modulo 2^k if used in concepts such as $A = A'$.

To this end, let $w = \sigma_0 \dots \sigma_{k-1} \in \Sigma^*$ be an arbitrary input word given to $\mathfrak{M}$. We next construct a TCQ $\Phi_{\mathfrak{M},w}$ and a TKB $\langle \mathcal{O}_{\mathfrak{M},w}, (\mathcal{A}_0) \rangle$ in *DL-Lite*$_{krom}$ such that $\mathfrak{M}$ accepts w iff $\phi_{\mathfrak{M},w}$ is satisfiable w.r.t. $\langle \mathcal{O}_{\mathfrak{M},w}, (\mathcal{A}_0) \rangle$.

Figure 7.2 illustrates the approach by showing an example representation of (parts of) computations of such an ATM. We use two counters modulo 2^k, A and A'. The tree describes computations in that one path describes one computation. The individual configurations are represented explicitly, one after the other, and each as a chain, such that every tree node represents one of the 2^k tape cells of a configuration; these cells are numbered by the A counter *invariant to time*. Each tree node or cell is represented by an individual in the reduction and, since these individuals are related by rigid roles, the computation tree "exists" at all time points; the time points are numbered by the A' counter. Branching models the universal transitions. Different from usual computation trees, the tree however splits at the node representing the cell under the head of the machine; the remaining parts of the configuration where the splitting occurs are replicated in each of the subtrees. The following example details the synchronization of the configurations.

Example 7.10 By means of Figure 7.2, we exemplarily describe the modeling of the transitions and corresponding successor configurations. In particular, the ontology encodes the *propagation* of the new state q and symbol σ of a transition, and of the cell

contents ϱ (of the considered configuration) that do not change to the successor configuration following in the tree. To represent all this information, we use corresponding *rigid* concept names and then propagate them via *flexible* concept names (marked by a prime) using the temporal dimension. Specifically, the tape contents of cell i are propagated to the successor configuration if the individual that represents the cell satisfies $A = A'$ (e.g., ϱ_1 and ϱ_2 at time points 1 and 2, respectively) and the cell is not under the head (e.g., ϱ_0 is not propagated at time point 0). The cell under the head can be identified because it (i.e., the corresponding individual) satisfies a state symbol q. It thus determines the state of the considered configuration and, together with the symbol in the cell, the transitions to be considered. Given a universal state, for each such transition, the configurations have one branch identified by a specific concept of the form $T_{q_0,\sigma_0,M}$; for an existential state, there is only one branch. The concept $T_{q_0,\sigma_0,M}$ initiates the propagation of the symbol σ_0 to the successor configuration if $A = A' + 1$. In this way, the propagation stops correctly in the next configuration at the cell that has been previously under the head, which satisfies $A = A'$ (i.e., it is left of the cell that previously satisfied $T_{q_0,\sigma_0,M}$). The new state q_0 is similarly propagated, but the corresponding time point specifically depends on the direction of the move M. For a right move, the propagation happens if the individual satisfying $T_{q_0,\sigma_0,M}$ also satisfies $A = A'$ and thus stops at the same cell in the successor configuration (recall that the individual satisfying $T_{q_0,\sigma_0,M}$ represents the cell right of the head) (e.g., see the propagation of q_0 and q_1 at time point 1). For left moves, the propagation should stop two cells left of the individual satisfying a concept of the form $T_{q_3,\sigma_3,L}$; we thus require the individual to satisfy $A = A' + 2$ to start the propagation. ♢

Before specifying the TCQ and ontology, we introduce all symbols we use below:

- A single named individual a identifies the root of the tree.
- Rigid role names $R_{q,\varrho,M}$ where $q \in \mathcal{Q}$, $\varrho \in \Gamma$, and $M \in \{L, R\}$ represent the edges of the tree. We collect all these role names in the set $\mathcal{R}$.

  Note that these roles represent the major difference to the reduction of [BGL12], where a single rigid role fulfills this purpose, but is used within qualified existential restrictions on the right-hand side of GCIs.
- Rigid concept names $A_0, \dots, A_{k-1}$ are used to model the bits of a binary counter numbering the tape cells in the configurations.
- Rigid concept names I and H point out special cells. In particular, I is satisfied by the nodes representing the initial configuration, and H is satisfied by all nodes representing a tape cell that is located (anywhere) to the right of the head in the current configuration.
- A rigid concept name, for each element in $\mathcal{Q} \cup \Gamma$, represents the tape content, the current state, and the head position in each configuration in the tree: if $\mathfrak{M}$ is in a state q and the head is on the i-th tape cell, then the individual (tree node) representing this cell satisfies the concept name q; we correspondingly represent the symbols in Γ.

- The rigid concept names $T_{q,\varrho,M}$, for all $q \in \mathcal{Q}$, $\varrho \in \Gamma$, and $M \in \{L, R\}$, are satisfied by an individual, representing a cell, if the head is on the left neighboring cell and the ATM executes the transition (q, σ, M) in the described configuration.

We use the temporal dimension to synchronize successor configurations in accordance with the chosen transition in order to model the change in the tape contents, the head position, and the state from one configuration to the next:

- Flexible concept names $A'_0, \dots, A'_{k-1}$ are used to model a counter in the temporal dimension. Its value is incremented (modulo 2^k) dual to the counter $A_0, \dots, A_{k-1}$ but along time and, at every time point, all individuals share the value of this counter. It is used for the synchronization of successor configurations: if the A' counter has value i, then the symbol in the i-th tape cell of any configuration (where i is not the head position) is propagated to the i-th tape cell of its successor configuration. Similarly, the state is propagated from the cells c directly right of the head position, each pointing out a specific transition (via the symbols $T_{q,\varrho,M}$), to the corresponding cells of the successor configurations (i.e., these cells have the same position on the tape as c for right-moves and otherwise lie two to the left).

- We further use a flexible concept name, for each element in $\mathcal{Q} \cup \Gamma$, which as above is distinguished from the rigid version by a prime. Considering a fixed time point, these names are used for the propagation of the state q or cell content σ of a cell c to the corresponding cell in the successor configuration(s). This propagation happens via the right neighboring cells of that configuration, which then satisfy q' and σ', respectively, at the time point whose A'-counter corresponds to the A-counter at c.

We may further use concept names of the form $\overline{A}$ for given concept names A as detailed in Lemma 7.2.

In the remainder of the proof, we define the TCQ $\Phi_{\mathfrak{M},w}$ and the TKB $\langle \mathcal{O}_{\mathfrak{M},w}, (\mathcal{A}_0) \rangle$ by describing the conjuncts of $\Phi_{\mathfrak{M},w}$ and listing the GCIs contained in $\mathcal{O}_{\mathfrak{M},w}$. To enhance readability, we may use GCIs that are not in *DL-Lite*$_{krom}$, but can be transformed as described in the beginning of this section (see Figure 7.1 and Example 7.4)[3]. We first express the tree structure in general.

- We enforce all elements to have some successor except if they satisfy q_a or q_r. Since the only elements satisfying a symbol from $\mathcal{Q}$ are the ones representing the position of the head, the tree generation thus is only stopped if we meet a halting configuration:

$$\overline{q_a} \sqcap \overline{q_r} \sqsubseteq \bigsqcup_{R_\delta \in \mathcal{R}} \exists R_\delta.$$

  Using a big disjunction over all possible roles, we can correctly represent the nondeterminism of the machine.

[3] Note that GCIs of the form of some of the critical GCIs neither occur in Figure 7.1 nor in Example 7.4. However, the right-hand sides of those GCIs are conjunctions. For that reason, the GCIs can be replaced by several copies of the original ones, each containing only one of the conjuncts on the right-hand side; the left-hand side is not changed.

- The A-counter is incremented alongside the tree modulo 2^k and modeled using the following GCIs for all $i \in [0, k-1]$:

$$\bigsqcap_{0 \le j \le i} A_j \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . \overline{A}_i,$$
$$\bigsqcap_{0 \le j < i} A_j \sqcap \overline{A}_i \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . A_i,$$
$$\Big(\bigsqcup_{0 \le j < i} \overline{A}_j \Big) \sqcap A_i \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . A_i,$$
$$\Big(\bigsqcup_{0 \le j < i} \overline{A}_j \Big) \sqcap \overline{A}_i \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . \overline{A}_i.$$

  For example, if the bits $A_0, \ldots, A_i$ are all true in the current tape cell, then in the successor cell these bits are all false.

  We thus have described a sequence of configurations where we can address single tape cells in all the configurations using the A counter. The latter restarts every time it has reached $2^k - 1$, and thus with each new configuration.

  The counter is initialized with value 0 at a. Hence, all elements representing the first tape cell in some configuration in the tree satisfy the auxiliary concept name $C_{A=0}$, defined as follows:

$$C_{A=0} \equiv \overline{A}_0 \sqcap \ldots \sqcap \overline{A}_{k-1}.$$

  Below, we use additional concept names of the form $C_{A=i}$, for (polynomially many) different values i, which we assume to be defined similarly.

  We further add the assertion

$$C_{A=0}(a)$$

  to $\mathcal{A}_0$. Since the names $\overline{A}_0, \ldots, \overline{A}_{k-1}$ are rigid, this assertion must be satisfied at every time point.

  We now enforce basic conditions which help to ensure that the tree actually represents a successful computation of $\mathfrak{M}$ on w.

- To formulate these conditions, we use the rigid concept name H to identify the tape cells that are to the right of the head:

$$\Big(H \sqcup \bigsqcup_{q \in \mathcal{Q}} q \Big) \sqcap \overline{C}_{A=2^k-1} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . H.$$

  Thus, the propagation stops at tree levels whose elements represent the last cell in a configuration, since these elements satisfy $C_{A=2^k-1}$.

  Observe that the disjunction can be replaced by an auxiliary concept name C_{aux} to obtain a CI in *DL-Lite*$_{krom}$ if we additionally consider the following GCI (see Example 7.4):

$$H \sqcup \bigsqcup_{q \in \mathcal{Q}} q \sqsubseteq C_{aux}.$$

- There is only one head position per configuration:

$$H \sqsubseteq \bigsqcap_{q \in \mathcal{Q}} \overline{q}.$$

  Note that we do not have to consider the elements representing the cells left to the head since, if such a cell satisfies a concept name from $\mathcal{Q}$, then all its successors in the tree are enforced to satisfy H.

- Each tape cell is associated with at most one state (which, at the same time, represents the position of the head):

$$\top \sqsubseteq \bigsqcap_{q_1, q_2 \in \mathcal{Q}, q_1 \neq q_2} \overline{q_1} \sqcup \overline{q_2}.$$

- Each tape cell contains exactly one symbol:

$$\top \sqsubseteq \bigsqcup_{\sigma \in \Gamma} \Big(\sigma \sqcap \bigsqcap_{\sigma' \in \Gamma \setminus \{\sigma\}} \overline{\sigma'} \Big).$$

  Observe that we obtain a GCI in *DL-Lite*$_{krom}$ if auxiliary names are introduced for the conjunctions on the right-hand side, in the way described above.

Before specifying the remaining, more intricate conditions for the synchronization of the configurations, we describe the first configuration in the tree (starting at a) as the initial configuration.

- In particular, we mark the corresponding elements by adding the assertion $I(a)$ to $\mathcal{A}_0$ and by propagating the concept alongside the first configuration as follows:

$$I \sqcap \overline{C}_{A=2^k-1} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . I.$$

- The first configuration is modeled by adding the assertion $q_0(a)$ to $\mathcal{A}_0$ and by considering the following GCIs for all $i \in [0, k-1]$:

$$I \sqcap C_{A=i} \sqsubseteq \sigma_i,$$
$$I \sqcap C_{A=k} \sqsubseteq B,$$
$$I \sqcap B \sqcap \overline{C}_{A=2^k-1} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . B$$

  where $w = \sigma_0 \dots \sigma_{k-1}$ is the input word.

We finally come to the most involved part, the synchronization of the configurations, which includes the modeling of the transitions.

- We first introduce the A'-counter, which is incremented along the temporal dimension. For every possible value of this counter, there is a time point where a

belongs to the concepts from the corresponding subset of $\{A'_0, \ldots, A'_{k-1}\}$. This is expressed using the following conjunct of $\Phi_{\mathfrak{M},w}$:

$$\Box_F \bigwedge_{0 \le i < k} \Big(\Big(\bigwedge_{0 \le j < i} A'_j(a) \Big) \leftrightarrow \big(A'_i(a) \leftrightarrow \bigcirc_F \neg A'_i(a) \big) \Big).$$

This formula expresses that the i-th bit of the A'-counter is flipped from one world to the next iff all preceding bits are true. Thus, the value of the A'-counter at the next world is equal to the value at the current world incremented by one.

Note that it is not necessary to initialize this counter to 0 in $\mathcal{A}_0$; we only need to know that all possible counter values are represented at some time point.

- The value of the A'-counter is always shared by all individuals:

$$\Box_F \Big(\bigwedge_{0 \le i < k} \exists x.A'_i(x) \to \neg \exists x.\overline{A}'_i(x) \Big).$$

For the application of the A'-counter, we introduce the abbreviation $E_{A,A'}$ describing the equality of the two counters:

$$E^i_{A,A'} \equiv (A_i \sqcap A'_i) \sqcup \left(\overline{A}_i \sqcap \overline{A}'_i \right),$$
$$E_{A,A'} \equiv \bigsqcap_{0 \le i < k} E^i_{A,A'}.$$

Furthermore, we define similar abbreviations as follows:

$$E_{A,(A'+1) \bmod 2^k} \equiv \bigsqcap_{0 \le j < k} \left(A'_j \sqcap \overline{A}_j \right) \sqcup \bigsqcup_{0 \le i < k} \left(\bigsqcap_{0 \le j < i} \left(A'_j \sqcap \overline{A}_j \right) \sqcap \overline{A}'_i \sqcap A_i \sqcap \bigsqcap_{i+1 \le j < k} E^j_{A,A'} \right),$$

$$E_{A,(A'+2) \bmod 2^k} \equiv E^0_{A,A'} \sqcap \left(\bigsqcap_{1 \le j < k} \left(A'_j \sqcap \overline{A}_j \right) \sqcup \bigsqcup_{1 \le i < k} \left(\bigsqcap_{1 \le j < i} \left(A'_j \sqcap \overline{A}_j \right) \sqcap \overline{A}'_i \sqcap A_i \sqcap \bigsqcap_{i+1 \le j < k} E^j_{A,A'} \right) \right).$$

We now can use the temporal dimension to propagate information from one level of the tree to the next one as outlined above, and hence specify the transitions.

- Symbols not under the head are copied:

$$\sigma \sqcap \bigsqcap_{q \in \mathcal{Q}} \overline{q} \sqcap E_{A,A'} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta.\sigma',$$
$$\sigma' \sqcap \overline{E}_{A,A'} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta.\sigma',$$
$$\sigma' \sqcap E_{A,A'} \sqsubseteq \sigma.$$

- To describe the transitions, we explicitly store chosen transitions with the help of the rigid concepts $T_{p,\varrho,M}$, by enforcing them to be satisfied by the elements representing the cells directly right-neighbored to the head position. Recall that there may be several such cells; we are now at the point where we specify the branching of the tree. Hence, we model the transitions for all $q \in \mathcal{Q}$ and $\sigma \in \Gamma$ using the following GCIs:

$$q \sqcap \sigma \sqsubseteq \bigsqcup_{\delta \in \Delta(q,\sigma)} \exists R_\delta, \quad \text{if } q \in \mathcal{Q}_\exists,$$
$$q \sqcap \sigma \sqsubseteq \bigsqcap_{\delta \in \Delta(q,\sigma)} \exists R_\delta, \quad \text{if } q \in \mathcal{Q}_\forall,$$
$$q \sqcap \sigma \sqsubseteq \bigsqcap_{\delta \in \Delta(q,\sigma)} \forall R_\delta . T_\delta .$$

  Observe that our main adaptation of the original proof is that we, instead of considering a single role R, regard all those in $\mathcal{R}$ and, instead of considering one R-successor per successor configuration δ, consider an R_δ-successor. This enables us to express a qualified existential restriction of the form $\bigsqcap_{\delta \in \Delta(q,\sigma)} \exists R.T_\delta$ on the right-hand side of a CI in the original proof, via the last two of the above CIs.[4]

- The (possible) replacement of the symbols under the head is described with the help of the transition concepts $T_{q,\sigma,M}$ for all $q \in \mathcal{Q}$, $\sigma \in \Gamma$, and $M \in \{L, R\}$:

$$T_{q,\sigma,M} \sqcap E_{A,(A'+1) \bmod 2^k} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . \sigma' .$$

  Recall that the transition concepts are only enforced to hold at the cell to the right of the current head position (hence the $+1$).

- The state information is similarly propagated for all $q \in \mathcal{Q}$ and $\sigma \in \Gamma$ as follows:

$$T_{q,\sigma,R} \sqcap E_{A,A'} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . q',$$
$$T_{q,\sigma,L} \sqcap E_{A,(A'+2) \bmod 2^k} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . q',$$
$$q' \sqcap \overline{E}_{A,A'} \sqsubseteq \bigsqcap_{R_\delta \in \mathcal{R}} \forall R_\delta . q',$$
$$q' \sqcap E_{A,A'} \sqsubseteq q.$$

We lastly enforce the computation to be an accepting one by disallowing the state q_r entirely using the GCI $q_r \sqsubseteq \bot$. Note that this is correct since we assume all the computations of $\mathfrak{M}$ to be terminating. This finishes the definition of the Boolean TCQ $\Phi_{\mathfrak{M},w}$ and the global ontology $\mathcal{O}_{\mathfrak{M},w}$—which is easily transformable into *DL-Lite*$_{krom}$—, which consist of the conjuncts and GCIs specified above. We further collect all assertions in the

[4] Recall that qualified existential restriction on the right-hand side of CIs can be modeled in *DL-Lite* if role inclusions are allowed (see the beginning of Chapter 6).

ABox $\mathcal{A}_0$. Given our descriptions above, it is easy to see that the size of $\Phi_{\mathfrak{M},w}$, $\mathcal{O}_{\mathfrak{M},w}$, and $\mathcal{A}_0$ is polynomial in k. Moreover, it can readily be checked that our constructions are equivalent to those in the proof of [BGL12, Thm. 4.1]. Hence, we get that $\Phi_{\mathfrak{M},w}$ is satisfiable w.r.t. $\langle \mathcal{O}_{\mathfrak{M},w}, (\mathcal{A}_0) \rangle$ iff $\mathfrak{M}$ accepts w. □

7.2 Data Complexity

Regarding data complexity, the CO-NP lower bound follows from CO-NP-hardness of conjunctive query answering w.r.t. *DL-Lite*$_{krom}$ knowledge bases. The latter is a consequence of [Cal+07b, Theorem 48 (1)], where the hardness is stated for *DL-Lite*$_{core}$ extended by CIs that allow for a concept $\neg A$, $A \in \mathsf{N_C}$, on the left-hand side. We reduce TCQ entailment in *DL-Lite*$^{\mathcal{H}}_{bool}$ to TCQ entailment in $\mathcal{ALCH}$, and then apply the results from [BBL15b] to obtain containment in CO-NP for the case where $\mathsf{N_{RR}} = \emptyset$, and in ExpTime. Note, however, that the case with rigid role names is still open for $\mathcal{ALCH}$ w.r.t. data complexity, which means that the latter result is not tight. This also leaves us a gap between CO-NP and ExpTime.

We assume $\mathcal{K}$ to be written in *DL-Lite*$^{\mathcal{H}}_{bool}$ and construct an $\mathcal{ALCH}$ TKB $\mathcal{K}'$ and a TCQ Φ' such that $\mathcal{K} \models \Phi$ iff $\mathcal{K}' \models \Phi'$. Note that the reduction leads to an exponential blowup in the size of the query, but this is irrelevant regarding data complexity. First, we extend the set of role names to include all inverse roles R^-, where R or R^- occurs in $\mathcal{O}$. Then, we construct the TKB $\mathcal{K}' := \langle \mathcal{O}', (\mathcal{A}'_i)_{0 \leq i \leq n} \rangle$ based on $\mathcal{K}$ by replacing all occurrences of concepts of the form $\exists R$, $R \in \mathsf{N}^-_{\mathsf{R}}$, by $\exists R.\top$, and adding the following axioms:

(i) a GCI $\exists R.(\neg \exists R^-.\top) \sqsubseteq \bot$ for each $R \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O})$,

(ii) an RI $R^- \sqsubseteq S^-$ for each $R \sqsubseteq S \in \mathcal{O}$.

We call a CQ φ' a *variant* of a CQ φ if φ' is obtained from φ by replacing some role atoms $R(s,t) \in \varphi$ by $R^-(t,s)$. We construct Φ' by replacing every CQ φ in Φ by a disjunction of all variants of φ. The correctness of this reduction is established next.

Lemma 7.11 *We have $\mathcal{K} \models \Phi$ iff $\mathcal{K}' \models \Phi'$.*

Proof. ($\Leftarrow$) Let $\mathfrak{I} = (\mathcal{I}_i)_{i \geq 0}$ be a model of $\mathcal{K}$ such that $\mathfrak{I} \not\models \Phi$. We show that we then have a model $\mathfrak{I}' = (\mathcal{I}'_i)_{i \geq 0}$ of $\mathcal{K}'$ such that $\mathfrak{I}' \not\models \Phi'$. Specifically, $\mathfrak{I}'$ has the same domain as $\mathfrak{I}$, interprets all symbols occurring in $\mathcal{K}$ as $\mathfrak{I}$ does, and, for all $\mathcal{I}'_i$, the interpretation of role names R^- such that $R^- \in \mathsf{N_R}(\mathcal{O}') \setminus \mathsf{N_R}(\mathcal{O})$, is equal to $(R^-)^{\mathcal{I}_i}$.

Given this definition of $\mathfrak{I}'$, we obviously have that $\mathcal{I}'_i \models \mathcal{A}'_i$ for all $i \in [0,n]$, since $\mathcal{I}_i \models \mathcal{A}_i$. The same holds for the GCIs and RIs that are contained in $\mathcal{O}$. Moreover, it is easy to see that the new GCIs and RIs are satisfied, too. We thus have $\mathfrak{I}' \models \mathcal{K}'$.

We now assume that $\mathfrak{I}' \models \Phi'$, by contradiction. Given the construction of Φ', we first show that, for every CQ φ in Φ which is replaced by a disjunction α in Φ', $\mathcal{I}'_i \models \alpha$ leads to $\mathcal{I}_i \models \varphi$, for all $i \geq 0$. Let thus π be a homomorphism of some CQ φ' in an arbitrary such disjunction α into some $\mathcal{I}'_i$, and let φ be the CQ that was replaced by α. φ and φ' thus only differ in the role atoms. Let $R(s,t)$ be an atom in φ and let $R^-(t,s)$ be the corresponding replacement in φ'. Then, we have $(\pi(s), \pi(t)) \in R^{\mathcal{I}_i}$, by

construction, which yields that π is also a homomorphism of φ into $\mathcal{I}_i$. Considering the other direction, we trivially have that $\mathcal{I}'_i \not\models \alpha$ leads to $\mathcal{I}_i \not\models \varphi$ for all $i \geq 0$, given our construction. By induction on the shape of Φ, it now can be easily shown that $\mathfrak{I} \models \Phi$ follows, which contradicts the assumption.

($\Rightarrow$) Let now $\mathfrak{I}' = (\mathcal{I}'_i)_{i\geq 0}$ be a model of $\mathcal{K}'$ such that $\mathfrak{I}' \not\models \Phi'$. We show this direction similarly by constructing a model $\mathfrak{I} = (\mathcal{I}_i)_{i\geq 0}$ of $\mathcal{K}$ such that $\mathfrak{I} \not\models \Phi$. In particular, we assume $\mathfrak{I}$ to have the same domain as $\mathfrak{I}'$, to interpret all concept names as $\mathfrak{I}'$ does, and to interpret all role names $R \in \mathsf{N_R}(\mathcal{O})$ such that $R^{\mathcal{I}_i} = R^{\mathcal{I}'_i} \cup \{(d,e) \mid (e,d) \in (R^-)^{\mathcal{I}'_i}\}$ for all $i \geq 0$.

The latter definition yields that $(d,e) \in R^{\mathcal{I}_i}$ if $(d,e) \in R^{\mathcal{I}'_i}$, and $(d,e) \in (R^-)^{\mathcal{I}_i}$ if $(d,e) \in (R^-)^{\mathcal{I}'_i}$ for all $i \geq 0$. Together with (i) and the definition of $R^{\mathcal{I}_i}$, we thus obtain that $e \in (\exists R)^{\mathcal{I}_i}$ iff $e \in (\exists R)^{\mathcal{I}'_i}$ for all $R \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O})$. Then, it is easy to see that we get $\mathcal{I}_i \models \mathcal{A}_i$ for all $i \in [0,n]$, since $\mathcal{I}'_i \models \mathcal{A}_i$. The above observation about concepts of the form $e \in (\exists R)^{\mathcal{I}_i}$, $R \in \mathsf{N}^-_{\mathsf{R}}(\mathcal{O})$, and the fact that the interpretations $\mathcal{I}'_i$ satisfy $\mathcal{O}$ further yields that $\mathcal{I}_i$ is a model of all CIs in $\mathcal{O}$. Lastly, we consider an arbitrary RI $R \sqsubseteq S$ in $\mathcal{O}$ and $(d,e) \in R^{\mathcal{I}_i}$. If $(d,e) \in R^{\mathcal{I}'_i}$, then $\mathcal{I}'_i \models \mathcal{O}$ implies $(d,e) \in S^{\mathcal{I}'_i} \subseteq S^{\mathcal{I}_i}$. Otherwise, we must have $(e,d) \in (R^-)^{\mathcal{I}'_i}$ and, by (ii) (i.e., we have $R^- \sqsubseteq S^- \in \mathcal{O}'$) and $\mathcal{I}'_i \models \mathcal{O}'$ get $(e,d) \in (S^-)^{\mathcal{I}'_i}$. But then, we have $(d,e) \in S^{\mathcal{I}_i}$, as well. Hence, we have $\mathfrak{I} \models \mathcal{K}$.

As above, we now assume that $\mathcal{I} \models \Phi$, by contradiction. Given the construction of Φ', we regard an arbitrary CQ φ in Φ replaced by a TCQ α in Φ' (i.e., α is a disjunction of CQs). If $\mathcal{I}' \models \alpha$, then there must be a homomorphism from a disjunct φ' of α into $\mathcal{I}'$, and thus obviously $\mathcal{I} \models \varphi$ by the constructions of α and $\mathcal{I}$. Let now π be a homomorphism of φ into $\mathcal{I}$. Then, for every role atom $R(s,t)$ in φ, we must either have $(\pi(s),\pi(t)) \in R^{\mathcal{I}'}$ or $(\pi(t),\pi(s)) \in (R^-)^{\mathcal{I}'}$, by the construction of $\mathcal{I}$. But then, π is also a homomorphism of the variant φ' contained in α that contains the corresponding combination of role atoms into $\mathcal{I}'$. This yields $\mathcal{I}' \models \alpha$. By induction, it now can be easily shown that $\mathcal{I}' \models \Phi'$ follows, which contradicts the assumption. □

This reduction allows us to use the results for TCQ entailment in $\mathcal{SHQ}$ [BBL15b, Thm. 5.2 and 4.15] to show the following.

Corollary 7.12 *TCQ entailment in DL-Lite$^{\mathcal{H}}_{bool}$ w.r.t. data complexity is*

- *in* co-NP *if* $\mathsf{N_{RR}} = \emptyset$*, even if* $\mathsf{N_{RC}} \neq \emptyset$*, and*
- *in* ExpTime*, even if* $\mathsf{N_{RR}} \neq \emptyset$*.*

We have thus completely classified TCQ entailment regarding ontologies in logics between *DL-Lite$_{krom}$* and *DL-Lite$^{\mathcal{H}}_{bool}$*, and different settings with rigid symbols. In summary, we have shown that the combined complexity between ExpTime and 2-ExpTime is not lower than that given for $\mathcal{ALC}$ or more expressive DLs [BBL15b; BBL15a]. In particular, we have 2-ExpTime-completeness if role inclusions are allowed, even if rigid symbols are disregarded. Also our data complexity results are equal to those for more expressive DLs. Hence, the case for data complexity where $\mathsf{N_{RR}} \neq \emptyset$ is an interesting open problem since it may still reveal an important difference between the expressive *DL-Lite* logics and even more expressive DLs.

8 Temporal Query Answering Without Negation

In this chapter, we propose a generic approach targeting the practical application of temporal ontology-based data access. We focus on a very general setting, based on the lessons learned in the previous chapters. The goal is to *answer temporal queries by rewriting* into standard query languages and, as the title suggests, this is achieved by dropping the negation operator (i.e., these rewritings are defined analogously to the first-order rewritings from Definition 2.22). We here focus on general query answering instead of only "yes/no" questions as considered in the previous chapters and, instead of investigating complexity, we show that temporal query answering can be rewritten; this allows for implementation based on existing tooling. Furthermore, we neither restrict the investigations to the $\mathcal{QL}$ queries introduced in Chapter 3 nor to particular DLs but also consider other query languages and logics that satisfy certain properties as ontology languages. These properties are satisfied by many fragments of *existential rules* (or simply *rules*) [Bag+11b]. Existential rules are function-free first-order implications where both the premise and conclusion are conjunctions of atomic formulas, and existentially quantified variables may occur only in the conclusion;[1] note that we allow for equality predicates. It is well known that query answering w.r.t. existential rules is undecidable in general [CLM81, Thm. 5.1][BV81, Thm. 7]. Yet, first decidable fragments have been proposed in the early days of database research and, still today, there is active research on such logics. The DLs $\mathcal{EL}$ and $\mathit{DL\text{-}Lite}_{horn}^{\mathcal{H}}$, for which TCQ entailment is investigated in Chapters 5 and 6 represent examples. Moreover, several other decidable rule fragments that meet our requirements and for which temporal query answering is thus rewritable are important in practice and successfully applied: examples are ontology languages proposed for information integration in databases, such as global-as-view mappings (e.g., see Figure 1.2); Datalog, a popular language for defining recursive views [AHV95]; the DLs $\mathcal{EL}^{++}$ [BBL05] and $\mathit{DL\text{-}Lite}_R$ [Cal+07b], which represent the logics behind profiles of the Web Ontology Language OWL 2; and many other description logics.

In what follows, we first generally specify these logics we focus on and describe several examples of concrete formalisms in Section 8.1,[2] then define the temporal query language in a similarly abstract way in Section 8.2, and finally prove the rewritability result in Section 8.3.

8.1 Preliminaries

In this section, we specify the syntax and semantics of the logics and (atemporal) queries we consider, to establish the logical framework for querying atemporal knowledge bases

[1] Alternative notions used in literature are, amongst others, $\forall\exists$-*rules* [Bag+11a], $\mathit{Datalog}^{\pm}$ [CGL09], and *tuple generating dependencies* [BV84].

[2] Note that, in line with the previous chapters, we primarily consider description logics.

in these logics. Note that these definitions are nearly analogous to but abstract from the ones given in Chapters 2 and 3. Thereafter, we detail the properties we require the formalisms to satisfy particularly. We also give a wealth of examples of concrete query formalisms that have been proposed in the literature and meet these requirements.

8.1.1 Logics

The basic setting is that of function-free first-order languages and focuses on signatures $\Sigma = (\Omega, (\Pi^n)_{n\geq 0})$ based on *constant symbols* Ω (also *constants*) and a family $(\Pi^n)_{n\geq 0}$ of sets of *n-ary predicate symbols* (also *predicates*). To simplify presentation, we assume the sets Ω and $\bigcup_{n\geq 0} \Pi^n$ to be non-empty and finite. Note that this assumption is reasonable given that a domain of interest usually focuses on certain symbols. We further assume that the sets of constants and predicate symbols are disjoint.

In our context, it is essential that the ground data is given separately from the other logical information since it may vary over time. We therefore focus on knowledge bases similar to those considered in the previous chapters and explicitly discern *facts* and *theories*. The latter represent the ontologies and may contain axioms more complex than facts. Since we introduce them in a very general sense, together with their semantics, we first specify the basis for model-theoretic semantics, *interpretations*.

Definition 8.1 (Semantics) An *interpretation* for a signature $\Sigma = (\Omega, (\Pi^n)_{n\geq 0})$ of FOL is a pair $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a non-empty set, the *domain* of $\mathcal{I}$, and $\cdot^\mathcal{I}$ is an *interpretation function* that assigns to every $c \in \Omega$ an element $c^\mathcal{I} \in \Delta^\mathcal{I}$ and, for all $n \geq 0$, to every $P \in \Pi^n$ an n-ary relation $P^\mathcal{I} \subseteq (\Delta^\mathcal{I})^n$.

Such an interpretation is called *finite* if its domain is finite. Two interpretations are *isomorphic* if there is a bijective mapping between their domains that preserves the interpretations of all constants and predicate symbols. ♢

The set of all interpretations is denoted by $\mathbb{I}$. In what follows, the signature of an interpretation is generally not mentioned explicitly if it is irrelevant or clear from the context.

We use the notion of *facts* as usual.

Definition 8.2 (Facts) Let $\Sigma = (\Omega, (\Pi^n)_{n\geq 0})$ be a first-order signature. A *fact* is an expression of the form $P(c_1, \ldots, c_n)$ where $P \in \Pi^n$ and $c_1, \ldots, c_n \in \Omega$.

An interpretation $\mathcal{I}$ is a *model* of a fact $P(c_1, \ldots, c_n)$, written $\mathcal{I} \models P(c_1, \ldots, c_n)$, if $(c_1^\mathcal{I}, \ldots, c_n^\mathcal{I}) \in P^\mathcal{I}$. ♢

Theories in a specific logical formalism are generally finite sets of axioms. For covering various logics that may focus on different kinds of axioms, we do not consider the nature of the axioms in more detail (i.e., apart from the facts already introduced). Specifically, we consider a generic *logic* that consists of the set of theories expressible in it and a satisfaction relation specifying the semantics.

Definition 8.3 (Logic) Let Σ be a first-order signature. A *logic* is a pair $(\mathcal{L}, \models_\mathcal{L})$, where $\mathcal{L}$ is a set of $\mathcal{L}$ *theories* over Σ and $\models_\mathcal{L} \subseteq \mathbb{I} \times \mathcal{L}$ is a *satisfaction relation* between interpretations and these $\mathcal{L}$ theories.

An interpretation $\mathcal{I}$ is a *model* of an $\mathcal{L}$ theory $\mathcal{T}$, written $\mathcal{I} \models_\mathcal{L} \mathcal{T}$, if $(\mathcal{I}, \mathcal{T}) \in \models_\mathcal{L}$. ♢

We define logics as sets of theories instead of sets of axioms since some logics put further restrictions on the shape of their theories apart from considering them to be sets of axioms (e.g., *sticky sets* are a fragment of Horn rules based on such a kind of restriction [CGP12]). Nevertheless, in many concrete logics, the basic satisfaction relation for axioms is lifted in a natural way to theories.

In the following, we often refer to a logic by its first component $\mathcal{L}$ and assume it to be implicitly associated with an entailment relation $\models_{\mathcal{L}}$. If the logic is clear from the context, we may also write $\models$ instead of $\models_{\mathcal{L}}$ and simply consider *theories*. As in the previous chapters, we focus on knowledge bases in the atemporal setting.

Definition 8.4 (Knowledge Base) Let $\mathcal{L}$ be a logic. A *knowledge base* (KB) written in $\mathcal{L}$ is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is an $\mathcal{L}$ theory and $\mathcal{A}$ is a finite set of facts, the *fact base*.

An interpretation $\mathcal{I}$ is a *model* of a fact base $\mathcal{A}$, written $\mathcal{I} \models \mathcal{A}$, if $\mathcal{I}$ is a model of all facts contained in $\mathcal{A}$. A knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is *consistent* if there is an interpretation that is a model of both $\mathcal{T}$ and $\mathcal{A}$. ♢

A basic requirement for the logics we consider is that consistency must be decidable. Note that consistency checking is generally one of the first steps in reasoning algorithms, since an inconsistent knowledge base makes most reasoning problems trivial.

The instances of our framework we focus on are fragments of existential rules. The point is that these rules do not allow to express disjunction—this is important for the conditions they have to satisfy. We close this part by providing concrete examples of such formalisms.

Example 8.5 Several Horn description logics extending $\mathcal{EL}$ and *DL-Lite*$^{\mathcal{H}}_{horn}$ satisfy our requirements (see Sections 1.2 and 2.1 for basics about DLs; further details are given in [Baa+07]). As described in Chapter 2, DL theories are called *ontologies* and contain *concept inclusions* and *role inclusions* as axioms. For most DLs, both of the latter can be expressed as rules. Horn DLs range from light-weight DLs such as $\mathcal{EL}$ and (some) members of the *DL-Lite* family to syntactically restricted forms of more expressive logics such as Horn-$\mathcal{SHIQ}$ [HMS07]. Also $\mathcal{EL}^{++}$ and *DL-Lite*$_R$, which represent the basis of the OWL 2 EL and QL profiles, are Horn DLs. In many Horn DLs, the inability to express disjunction leads to the interesting property that knowledge bases can be characterized in terms of a single *canonical model*.

Regarding KBs with theories specifying constraints such as global-as-view mappings, [DNR08] study similar *universal models*.[3] Given different applications of universal models in the literature, they observe that the universality of a model, the fact that there is a homomorphism into every model of a knowledge base, makes universal models apt for solving various reasoning problems, such as query answering.

Datalog extends global-as-view mappings. In particular, it allows for defining recursive views. The axioms are rules as introduced above with the restriction that every variable that occurs in the conclusion must also occur in the premise; thus, rules without premise are facts. Theories are finite sets of such rules and called *Datalog programs*. An interesting property of Datalog is that every program $\mathcal{P}$ has a *least Herbrand model*,

[3] [DNR08] also consider constraints that are more expressive than existential rules but remark that universal models do not generally exist in all these settings.

which satisfies exactly those facts that hold in all models of $\mathcal{P}$ (similar to the canonical models of knowledge bases in Horn DLs). Since we do not consider function symbols, the Herbrand domain is Ω, and thus the least Herbrand model is finite. In this work, we consider the *linear* Datalog fragment, where the premise of a rule may contain at most one atom that also occurs in the conclusion of some rule in the program [AHV95].

Theories of logics in the *Datalog*$^{\pm}$ family generalize Datalog programs in that they, next to the equality predicate and the truth constant false, most importantly, allow for existential quantification in the conclusion of rules [CGL12] (i.e., *Datalog*$^{\pm}$ represents a special kind of existential rules). Since already single such features (e.g., equality) lead to undecidability of most reasoning problems, the logics proposed in the literature impose rather strong restrictions on the shape of the theories. The *linear* fragment, where the rule premises may contain only one atom [CGL09], or the above mentioned sticky sets represent examples of such logics. ♢

8.1.2 Query Answering

The query languages we consider are also specified in a largely generic way and given together with their semantics.

Definition 8.6 (Query Language) Let $\Sigma = (\Omega, (\Pi^n)_{n\geq 0})$ be an FO signature and $\mathsf{N_V}$ be a set of *variables* disjoint from Ω and $(\Pi^n)_{n\geq 0}$. A *variable assignment* is a total mapping of the form $\mathfrak{a}\colon \{x_1, \ldots, x_n\} \to \Omega$, $x_1, \ldots, x_n \in \mathsf{N_V}$.

A *query language* is a triple $(\mathcal{QL}, \mathsf{N_{FV}}, \models_{\mathcal{QL}})$, where $\mathcal{QL}$ is a set of $\mathcal{QL}$ *queries* over Σ, $\mathsf{N_{FV}}\colon \mathcal{QL} \to 2^{\mathsf{N_V}}$ maps every $\mathcal{QL}$ query to the finite set of its *free variables*, and $\models_{\mathcal{QL}}$ is a *satisfaction relation*; for an interpretation $\mathcal{I}$, a $\mathcal{QL}$ query φ, and a variable assignment $\mathfrak{a}\colon \mathsf{N_{FV}}(\varphi) \to \mathsf{N_C}$[4], the latter is denoted by $\mathcal{I} \models_{\mathcal{QL}} \mathfrak{a}(\varphi)$, and it is such that the following hold, for all $\varphi \in \mathcal{QL}$:

(i) For all assignments $\mathfrak{a}_1, \mathfrak{a}_2\colon \{x_1, \ldots, x_n\} \to \mathsf{N_C}$ and interpretations $\mathcal{I}$ such that $\mathfrak{a}_1(x_i)^{\mathcal{I}} = \mathfrak{a}_2(x_i)^{\mathcal{I}}$ for all $i \in [1, n]$, we have $\mathcal{I} \models_{\mathcal{QL}} \mathfrak{a}_1(\varphi)$ iff $\mathcal{I} \models_{\mathcal{QL}} \mathfrak{a}_2(\varphi)$.

(ii) For all assignments $\mathfrak{a}\colon \{x_1, \ldots, x_n\} \to \mathsf{N_C}$ and isomorphic interpretations $\mathcal{I}$ and $\mathcal{J}$, we have $\mathcal{I} \models_{\mathcal{QL}} \mathfrak{a}(\varphi)$ iff $\mathcal{J} \models_{\mathcal{QL}} \mathfrak{a}(\varphi)$.

If $\mathcal{I} \models_{\mathcal{QL}} \mathfrak{a}(\varphi)$, then $\mathfrak{a}$ is an *answer* to φ w.r.t. $\mathcal{I}$. ♢

Condition (i) expresses that the satisfaction of queries in an interpretation does not depend on the names of constants in the queries but only on their interpretation, and Condition (ii) expresses that it is actually characterized by the interpretation. Note that we need these assumptions for technical reasons; though, they are reasonable in general.

We adopt the same conventions as for logics and refer to query languages only by their first component, assume the satisfaction relation to be implicitly associated to it, and denote that relation by $\models$ if $\mathcal{QL}$ is clear from the context. $\mathsf{Ans}(\varphi, \mathcal{I}) \subseteq \mathsf{N_C}^{\mathsf{N_{FV}}(\varphi)}$ denotes the set of all answers to a query φ w.r.t. an interpretation $\mathcal{I}$.

Answering queries w.r.t. single interpretations is however not our main goal. Instead, we focus on the answers w.r.t. a knowledge base, the so-called *certain answers*.

[4] We do not consider variable assignments that do not map exactly the free variables of the query.

Definition 8.7 (Certain Answer) Let $\mathcal{L}$ be a logic, $\mathcal{QL}$ a query language, $\mathcal{K}$ a knowledge base written in $\mathcal{L}$, and φ a $\mathcal{QL}$ query. A variable assignment $\mathfrak{a}\colon \mathsf{N_{FV}}(\varphi) \to \Omega$ is a *certain answer* to φ w.r.t. $\mathcal{K}$, written $\mathcal{K} \models \mathfrak{a}(\varphi)$, if $\mathfrak{a}$ is an answer to φ w.r.t. every model of $\mathcal{K}$. ♢

The reasoning problem in focus here is *query answering*: the task of computing the set of all certain answers to a query φ w.r.t. a knowledge base $\mathcal{K}$. The set of those answers is denoted by $\mathsf{Cert}(\varphi, \mathcal{K})$.

A special situation arises when the considered queries have no free variables. Queries of this form are called *Boolean* queries since the set $\mathsf{Cert}(\varphi, \mathcal{K})$ can only be empty or contain the empty variable assignment as its only element. In the latter case, φ is *entailed* by $\mathcal{K}$. Similarly, regarding an interpretation $\mathcal{I}$ and a Boolean query φ, $\mathcal{I} \models \varphi$ implies that $\mathsf{Ans}(\varphi, \mathcal{I})$ is not empty and, specifically, only contains the empty set.

We assume that every query language contains a special Boolean query *true*, which holds in all interpretations. Likewise, we assume the presence of a Boolean query *false*, which does not hold in any interpretation. These queries can be added to a query language without affecting any of the properties or constructions described in the following. We also conclude the introduction of the queries with concrete examples.

Example 8.8 The simplest query language arises from considering all facts as Boolean queries and, correspondingly, taking $\models_{\mathcal{L}}$ as $\models_{\mathcal{QL}}$; the variable assignments can be disregarded. The entailment of a fact by a knowledge base is then equivalent to the original definition.

Similarly, we can consider the Boolean query language $\mathcal{QL} := \mathcal{L}$ with $\models_{\mathcal{QL}}$ given by $\models_{\mathcal{L}}$. That is, we can ask for the entailment of theories. In the context of description logics, an important such query language is that of *subsumptions*, which ask whether single CIs are entailed (see Definition 2.5).

Instance queries (IQs) generalize fact queries by allowing for variables. That is, an IQ is an atom whose variables are considered as the free variables of the query. To compute $\mathsf{Cert}(\varphi, \mathcal{K})$ for an IQ φ and KB $\mathcal{K}$, all variable assignments that *certainly* (in all models of $\mathcal{K}$) make the fact true when replacing the free variables accordingly have to be determined.

For relational databases, an important class of queries are *conjunctive queries* (CQs) (also *select-project-join queries*) of the form $\exists y_1, \ldots, y_m.\varphi$, where $y_1, \ldots, y_m \in \mathsf{N_V}$ and φ is a conjunction of instance queries.[5] The free variables of a CQ are all those occurring in it except for $y_1, \ldots, y_m$. In contrast to the free variables, which range only over the constants, the existentially quantified variables range over the whole domain of a given interpretation. The semantics of CQs is obtained by viewing them as first-order sentences in the obvious way. In the standard database setting, which does not include a logical theory, the focus is on computing $\mathsf{Ans}(\varphi, \mathcal{I})$ for a conjunctive query φ and a *finite* interpretation $\mathcal{I}$, representing a given relational database. This is realized by formulating φ in a query language supported by the system (e.g., SQL is a standard for such a language) and evaluating it over the database. The more general problem of computing certain answers to conjunctive queries w.r.t. a knowledge base has been

[5] Note that we here generalize the definition of CQs given in Section 2.1.2 for the DL setting in that we consider more general signatures.

investigated for many logical formalisms, in particular DLs and rules [Cal+07b; Gli+08; LTW09; RG10].

Unions of conjunctive queries (UCQs) (i.e., disjunctions of CQs) play a role similarly important to the one of conjunctive queries. Moreover, many approaches for answering CQs w.r.t. an ontology are based on rewriting to UCQs (e.g., see Lemma 2.24).

Another class of interest between UCQs and arbitrary first-order queries are *positive existential queries* (PEQs) are of the form $\exists y_1, \ldots, y_m.\varphi$, where φ is a positive Boolean combination of instance queries (i.e., using conjunction and disjunction, but no negation).

Conjunctive regular path queries (CRPQs) are a popular query language for graph databases and are also studied in the context of description logics, where the predicates considered are at most binary. They generalize conjunctive queries in a different direction, by allowing conjuncts of the form $L(x, y)$, where L is a regular expression over the binary predicate symbols and x and y represent variables. If an interpretation over such a signature is regarded as a labeled graph, these conjuncts express the existence of a path from x to y such that the concatenation of its edge labels—DL role names—belongs to the language generated by L. *Conjunctive two-way regular path queries* (C2RPQs) extend CRPQs in that the regular expressions may use the binary predicates in the backwards direction.

Datalog queries are pairs of the form (P, G) consisting of a Datalog program P and a *goal predicate* G, which is to be answered over P. That is, Datalog query answering w.r.t. a fact base is similar to instance query answering w.r.t. a KB but, with Datalog, the theory is considered as a part of the query. The program P contains auxiliary predicates that are local to the query (i.e., the predicates are not part of a signature of an interpretation in which the query is to be answered) and used to evaluate it. Specifically, only auxiliary predicates are allowed to occur in the conclusions of rules, and the goal predicate must be an auxiliary predicate. A variable assignment $\mathfrak{a}$ is an answer to such a query in an interpretation $\mathcal{I}$ if all extensions of $\mathcal{I}$ to the auxiliary predicates that satisfy P also satisfy $G(\mathfrak{a}(x_1), \ldots, \mathfrak{a}(x_n))$. This is equivalent to the containment of this fact in the least Herbrand model of the KB $\langle P, \mathsf{facts}(\mathcal{I})\rangle$, where $\mathsf{facts}(\mathcal{I})$ denotes the (finite) set of all facts that $\mathcal{I}$ is a model of.

Note that every UCQ can be formulated as a Datalog query, by regarding the CQs as premises of rules and introducing a single, fresh goal predicate for the conclusions. Similarly, PEQs correspond to Datalog queries with *nonrecursive* programs [AHV95, Thm. 4.5.2 and 5.4.10]. ♢

8.1.3 Canonical Models and Rewritability

We finally provide the properties we assume the logics and queries we consider to satisfy next to the decidability of consistency. First, we focus on canonical models as mentioned above. We tailor the corresponding property however to the reasoning problem of query answering.

Definition 8.9 (Canonical Model Property) A logic $\mathcal{L}$ has the *canonical model property* w.r.t. a query language $\mathcal{QL}$ if, for each consistent knowledge base $\mathcal{K}$ written in $\mathcal{L}$, there is a countably infinite *canonical model* $\mathcal{I}_\mathcal{K}$ of $\mathcal{K}$ such that, for all $\mathcal{QL}$ queries φ, $\mathsf{Cert}(\varphi, \mathcal{K}) = \mathsf{Ans}(\varphi, \mathcal{I}_\mathcal{K})$. ♢

$\mathcal{L}$	$\mathcal{QL}$	shown in
$\mathcal{EL}^{++}$	subs.	[BBL05]
$\mathcal{ELH}$	UCQ	[Ros07, Lem. 1]
$\mathcal{ELI}^{f}$	CQ	[KL07, Lem. 5]
$\mathcal{ELH}_{\bot}^{dr}$	CQ	[LTW09, Prop. 4]
$\mathcal{ELHI}^{\neg}$	CQ	[PMH10, Lem. 10]
DL-Lite$_{[\mathcal{R}\|\mathcal{F}]}$	UCQ	[Cal+07b, Thm. 29]
DL-Lite$_{horn}^{(\mathcal{HN})}$	UCQ	[BAC10, Thm. 9]
DL-Lite$_{horn}$	PEQ	[Kon+11, Thm. 3]
DL-Lite$_{\mathcal{R}}$	C2RPQ	[BOS15, Lem. 3.2]
Horn-$\mathcal{ALCHIQ}$	CQ	[Eit+12, Thm. 3], [Kaz09]
Horn-$\mathcal{ALCHOIQ}_{\mathsf{Self}}^{\mathsf{Disj}}$	C2RPQ	[ORŠ11, Thm. 2]

Figure 8.1: DLs $\mathcal{L}$ and query languages $\mathcal{QL}$ w.r.t. which they have the canonical model property.

The restriction to countably infinite interpretations is again technical and ensures that all these models have the same cardinality. This is however only a minor restriction since canonical models are often explicitly constructed in a countable way. Moreover, in case the canonical model is finite, it can usually be extended by (countably infinite) replications without affecting the semantics (i.e., the answers to queries are the same w.r.t. both models). Example 8.10 presents logics, particularly DLs, that have this property.

Example 8.10 Figure 8.1 lists several DLs $\mathcal{L}$ and query languages $\mathcal{QL}$ that have the canonical model property. The canonical model is usually constructed by applying the axioms of the knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ as *completion rules* to the facts in $\mathcal{A}$ in order to obtain a model of $\mathcal{K}$.[6]

In [BBL05], the focus is on subsumption queries. A canonical model is not explicitly constructed, but it can be easily obtained by regarding the application of the completion rules (see Table 2 in that paper). This is also the case for [Kaz09] (see Table 3 in that paper).

Note that the result from [BOS15] also holds for $\mathcal{ELH}$. Further, the one from [Eit+12] also holds for the more expressive DL Horn-$\mathcal{SHIQ}$ if the CQs only contain *simple* roles (i.e., roles without transitive subroles).

Regarding a Datalog program obtained from $\mathcal{K}$ (i.e., the facts in $\mathcal{A}$ are regarded as rules with empty body), [PMH10; Eit+12] construct a canonical model based on the least Herbrand model of the program. ♢

Recently, so-called *rewriting approaches*—in the spirit of the rewriting from data integration (e.g., (1.2) in Chapter 1)—for computing the set of certain answers to a given query have become popular. They are usually based on a kind of canonical

[6] The procedure of applying the completion rules as well as the model are also called *chase* in the literature [DNR08].

model: the idea is to rewrite a query that is to be answered over a KB such that it can be evaluated over a single and *finite* interpretation. We specify a corresponding property of *rewritability* that is more general than the FO rewritability considered in Chapter 2 (see Definition 2.22).

Observe that the rewritten query, called *rewriting*, usually belongs to a more expressive query language. Next to the type of this language, rewriting approaches can be discerned regarding the information that is used for constructing the rewriting and the interpretation, respectively. *Strict* rewriting approaches construct the rewriting based on the theory and the original query, and consider the interpretation to be the (unmodified) fact base, interpreted under the closed-world assumption. Since this technique is considered to be very efficient, it is employed in many works (e.g., in [Cal+05; Cal+07b; CGL09]). More general approaches, such as the *combined approach* [Kon+10; Kon+11], are different in that they allow the interpretation to also be influenced by the theory. Note that, in practice, the data given then has to be preprocessed before query answering can be performed.

Definition 8.11 (Rewritable) Let $\mathcal{L}$ be a logic and $\mathcal{QL}_1$ and $\mathcal{QL}_2$ be query languages. Then, $\mathcal{QL}_1$ queries are $\mathcal{QL}_2$ *rewritable* w.r.t. $\mathcal{L}$ if the following can be computed:

- for every $\mathcal{QL}_1$ query φ and theory $\mathcal{T}$ written in $\mathcal{L}$, a $\mathcal{QL}_2$ query $\varphi^{\mathcal{T}}$ such that $\mathsf{N_{FV}}(\varphi) = \mathsf{N_{FV}}(\varphi^{\mathcal{T}})$;
- for every $\mathcal{L}$ theory $\mathcal{T}$, a finite set $\Delta_{\mathcal{T}}$ such that $\Omega \subseteq \Delta_{\mathcal{T}}$;
- for every consistent knowledge base $\mathcal{K}$ written in $\mathcal{L}$, a finite interpretation $\mathcal{D}_{\mathcal{K}}$ over the domain $\Delta_{\mathcal{T}}$ such that $c^{\mathcal{D}_{\mathcal{K}}} = c$ for all $c \in \Omega$ (*standard name assumption*);

such that, for all *consistent* knowledge bases $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ written in $\mathcal{L}$ and $\mathcal{QL}_1$ queries φ, we have $\mathsf{Cert}(\varphi, \mathcal{K}) = \mathsf{Ans}(\varphi^{\mathcal{T}}, \mathcal{D}_{\mathcal{K}})$. ♢

In summary, $\mathcal{QL}_2$ rewritability means that finding certain answers to $\mathcal{QL}_1$ queries w.r.t. $\mathcal{L}$ KBs can be reduced to finding (ordinary) answers to $\mathcal{QL}_2$ queries in finite interpretations. Our last requirement is therefore that the set of answers to a $\mathcal{QL}_2$ query w.r.t. a finite interpretation must be computable. This means that $\mathcal{QL}_2$ rewritability of $\mathcal{QL}_1$ queries w.r.t. $\mathcal{L}$ implies that the set of answers to a $\mathcal{QL}_1$ query w.r.t. a knowledge base is also computable. Focusing on data complexity, observe that Definition 8.11 does not only abstract from but also extend Definition 2.22 in order to capture more existing rewriting approaches. The latter definition regards a finite interpretation that is independent of the theory and the query. In contrast, we here require $\mathcal{D}_{\mathcal{K}}$ only to be independent of a concrete query. The rewriting $\varphi^{\mathcal{T}}$ must however still not depend on knowledge from a fact base. Definition 8.11 again includes technical restrictions that are needed to lift the atemporal approach to the temporal setting (i.e., the assumption that $\Delta_{\mathcal{T}}$ is independent of the data, and the standard name assumption). But all of them are satisfied by the logics and query languages described in the following example, for which rewritability is given.

Example 8.12 Figure 8.2 lists several rewritability results for different instances of $\mathcal{L}$, $\mathcal{QL}_1$, and $\mathcal{QL}_2$. For the logics of the *DL-Lite* family and the $\mathcal{EL}$ extensions, the finite interpretation $\mathcal{D}_{\mathcal{K}}$ is usually obtained by viewing the fact base under the closed world

$\mathcal{L}$	$\mathcal{QL}_1$	$\mathcal{QL}_2$	shown in
$\mathcal{EL}^{++}$	subs.	subs.	[BBL05]
$\mathcal{ELH}_{\bot}^{dr}$	CQ	$\text{FO}_=$	[LTW09, Thm. 5]
$\mathcal{ELHI}^{\neg}$	CQ	Datalog	[PMH10, Thm. 2, Lem. 16]
DL-Lite^{+}	CQ	UCQ^{+}	[PMH10, Thm. 2, Lem. 16]
DL-Lite$_{\mathcal{R}}$	CQ	UCQ	[Cal+07b, Lem. 39]
DL-Lite$_{\mathcal{R}}$	UCQ	PEQ	[RA10, Thm. 2]
DL-Lite$_{horn}^{\mathcal{N}}$	CQ	$\text{FO}_=$	[Kon+10, Thm. 10]
DL-Lite$_{horn}^{(\mathcal{HN})}$	UCQ	UCQ	[BAC10, Lem. 10]
DL-Lite$_{\mathcal{R}}$	C2RPQ	C2RPQ	[BOS15, Prop. 6.3]
DL-Lite^{+}	CQ	CRPQ	[Dim+16, Thm. 3]
Horn-$\mathcal{ALCHIQ}$	CQ	UCQ	[Eit+12, Thm. 4]
$\mathcal{LDL}^{+}$	IQ	IQ	[HEX10, Cor. 11]
$\mathcal{SROEL}(\sqcap, \times)$	IQ	IQ	[Krö11, Thm. 1]
$\text{Datalog}^{\pm}$ family	CQ	UCQ	[GOP11, Thm. 1]

Figure 8.2: Rewritability results for different instances of $\mathcal{L}$, $\mathcal{QL}_1$, and $\mathcal{QL}_2$; $\text{FO}_=$ denotes first-order queries with equality and UCQ^{+} a combination of a UCQ with a linear Datalog program.

assumption, but sometimes additional constant symbols are introduced (e.g., prototypical elements as in Definitions 2.10 and 2.11). In the other cases, $\mathcal{D}_{\mathcal{K}}$ is based on the least Herbrand model of a suitable Datalog program that is based on $\mathcal{K}$.

Again, the rewritability result from [BBL05] is only implicitly given in that paper (see Lemma 3).

As in Example 8.10, the result from [BOS15] also holds for $\mathcal{ELH}$; and the one from [Eit+12] also holds for Horn-$\mathcal{SHIQ}$ if the CQs do not contain non-simple roles.

In the constructions for $\mathcal{LDL}^{+}$ and $\mathcal{SROEL}(\sqcap, \times)$, the original query is not adapted, and therefore these logics also have the canonical model property.

To ensure the termination of the rewriting algorithm in [GOP11], the theories have to be restricted (e.g., to *linear* rules or sticky sets). ♢

Many works suggest to consider rewritability as a decision problem as follows: Given a logic $\mathcal{L}$, a $\mathcal{QL}_1$ query, and a query language $\mathcal{QL}_2$, decide if the query is $\mathcal{QL}_2$ rewritable w.r.t. $\mathcal{L}$ [Bie+14; CR15; Han+15; Bie+16]. The assumption behind this approach is that, in practice, concrete query answering problems for expressive logics are actually rewritable because many features of the logics are rarely used for modeling. Indeed, in the experiments of [Han+15], efficient[7] FO rewritings of instance queries in $\mathcal{ELH}^{dr}$ were obtained in most cases and turned out to be very performant. Based on these results, our approach extends to more expressive logics, too: instead of $\mathcal{QL}_1$, we can consider

[7] Many of the rewriting approaches proposed in the past are based on backward chaining and rewriting into UCQs, which requires a lot of optimization to obtain feasibility in applications.

only those elements of $\mathcal{QL}_1$ that have this property, and thus obtain another instance of Definition 8.11.

Lastly, note that finiteness of $\mathcal{D}_\mathcal{K}$ is obviously not sufficient in practice, where the interpretations $\mathcal{D}_\mathcal{K}$ additionally should be small, so that $\mathcal{QL}_2$ queries can be evaluated efficiently. Indeed, many rewritability results have subsequently been refined into that direction. However, in this final (technical) chapter, we investigate rewritability in a very general temporal setting and consider theoretical complexity considerations to be out of scope.

8.2 Positive Temporal Queries

In this section, we regard a logic $\mathcal{L}$ and a query language $\mathcal{QL}$ and lift the definitions of the previous section to a temporal setting corresponding to the one proposed in Section 3.1 for description logics and TQs.

In particular, we also consider *temporal knowledge bases* (TKBs) $\mathcal{K} = \langle \mathcal{T}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$—now, written in $\mathcal{L}$—built from global $\mathcal{L}$ theories and finite sequences of fact bases (see Definition 3.2). The semantics here is correspondingly based on sequences $\mathfrak{I} = (\mathcal{I}_i)_{i \geq 0}$ of FOL interpretations sharing one domain, which are called *temporal $\mathcal{L}$ structures*. The fact that $\mathfrak{I}$ is a *model* of $\mathcal{K}$ is similarly denoted by $\mathfrak{I} \models \mathcal{K}$ and holds if $\mathcal{I}_i \models_\mathcal{L} \mathcal{T}$ and $\mathcal{I}_i \models \mathcal{A}_i$ for all $i \in [0, n]$. A TKB is *consistent* if it has a model.

Positive temporal queries do not contain negation and are therefore defined more fine granularly than the temporal queries introduced in Definition 3.4, since we cannot consider $\Box_F$ and $\Box_P$ to be derived operators (see Figure 2.2). But we can consider $\Diamond_F$ and $\Diamond_P$ to be such operators because we assume *true* to be a $\mathcal{QL}$ query.

Definition 8.13 (Syntax of Positive Temporal Queries) Let $\Sigma = (\Omega, (\Pi^n)_{n \geq 0})$ be a FO signature. The set of *positive temporal $\mathcal{QL}$ queries* (PTQs) over Σ is defined by the following grammar:

$$\Phi ::= \varphi \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid \bigcirc_F \Phi_1 \mid \bigcirc_P \Phi_1 \mid \Box_F \Phi_1 \mid \Box_P \Phi_1 \mid \Phi_1 \,\mathcal{U}\, \Phi_2 \mid \Phi_1 \,\mathcal{S}\, \Phi_2$$

where φ is a $\mathcal{QL}$ query over Σ and Φ_1 and Φ_2 are positive temporal $\mathcal{QL}$ queries, in their turn. ♢

We in the following use the term *positive temporal queries* if $\mathcal{QL}$ is clear from the context. The set of *free variables* $\mathsf{FV}(\Phi)$ of a positive temporal $\mathcal{QL}$ query Φ is the union of the free variables of the $\mathcal{QL}$ queries in it, and PTQs for which this set is empty are *Boolean* PTQs. A query is a *subquery* of a PTQ Φ if it occurs in Φ and, if it is atemporal, occurs at least once not as part of another atemporal query in Φ. $\mathbb{S}(\Phi)$ denotes the set of all subqueries of Φ, including Φ itself. For a subquery Ψ of Φ, we denote by $\mathfrak{a}_\Psi$ the restriction of a variable assignment $\mathfrak{a} \colon \mathsf{FV}(\Phi) \to \Omega$ to $\mathsf{FV}(\Psi)$.

Since we here focus on query answering, we explicitly consider the free variables with the semantics.

Definition 8.14 (Semantics of Positive Temporal Queries) Let Φ be a PTQ, $\mathfrak{I} = (\mathcal{I}_i)_{i \geq 0}$ be a sequence of interpretations over a common domain, $\mathfrak{a} \colon \mathsf{FV}(\Phi) \to \Omega$ be a variable assignment, and $i \in [0, n]$ represent a time point. The *satisfaction relation* $\mathfrak{I}, i \models \mathfrak{a}(\Phi)$ is defined by induction on the structure of Φ as specified in Figure 8.3.

PTQ Φ	Condition for $\mathfrak{I}, i \models \mathfrak{a}(\Phi)$
$\mathcal{QL}$ query ψ	$\mathcal{I}_i \models \mathfrak{a}(\psi)$
$\Phi_1 \wedge \Phi_2$	$\mathfrak{I}, i \models \mathfrak{a}_{\Phi_1}(\Phi_1)$ and $\mathfrak{I}, i \models \mathfrak{a}_{\Phi_2}(\Phi_2)$
$\Phi_1 \vee \Phi_2$	$\mathfrak{I}, i \models \mathfrak{a}_{\Phi_1}(\Phi_1)$ or $\mathfrak{I}, i \models \mathfrak{a}_{\Phi_2}(\Phi_2)$
$\bigcirc_F \Phi_1$	$\mathfrak{I}, i+1 \models \mathfrak{a}(\Phi_1)$
$\bigcirc_P \Phi_1$	$i > 0$ and $\mathfrak{I}, i-1 \models \mathfrak{a}(\Phi_1)$
$\Box_F \Phi_1$	$\mathfrak{I}, k \models \mathfrak{a}(\Phi_1)$ for all k, $i \leq k$
$\Box_P \Phi_1$	$\mathfrak{I}, k \models \mathfrak{a}(\Phi_1)$ for all k, $0 \leq k \leq i$
$\Phi_1 \,\mathcal{U}\, \Phi_2$	there is a k, $i \leq k$, with $\mathfrak{I}, k \models \mathfrak{a}_{\Phi_2}(\Phi_2)$ and $\mathfrak{I}, j \models \mathfrak{a}_{\Phi_1}(\Phi_1)$ for all j, $i \leq j < k$
$\Phi_1 \,\mathcal{S}\, \Phi_2$	there is a k, $0 \leq k \leq i$, with $\mathfrak{I}, k \models \mathfrak{a}_{\Phi_2}(\Phi_2)$ and $\mathfrak{I}, j \models \mathfrak{a}_{\Phi_1}(\Phi_1)$ for all j, $k < j \leq i$

Figure 8.3: Semantics of temporal $\mathcal{QL}$ queries.

If $\mathfrak{I}, i \models \mathfrak{a}(\Phi)$, then $\mathfrak{a}$ is an *answer* to Φ w.r.t. $\mathfrak{I}$ at i. $\mathfrak{a}$ is a *certain answer* to Φ w.r.t. a TKB $\mathcal{K}$ at i, written $\mathcal{K}, i \models \mathfrak{a}(\Phi)$, if $\mathfrak{a}$ is an answer to Φ at i in every model of $\mathcal{K}$. $\diamondsuit$

The set of all answers to Φ w.r.t. $\mathfrak{I}$ at time point i is denoted by $\mathsf{Ans}(\Phi, \mathfrak{I}, i)$, and the set of all certain answers to Φ w.r.t. $\mathcal{K}$ is denoted by $\mathsf{Cert}(\Phi, \mathcal{K}, i)$. Since our focus is on the time point n, we introduce the abbreviations $\mathsf{Ans}(\Phi, \mathfrak{I}) := \mathsf{Ans}(\Phi, \mathfrak{I}, n)$ and $\mathsf{Cert}(\Phi, \mathcal{K}) := \mathsf{Cert}(\Phi, \mathcal{K}, n)$. A Boolean PTQ Φ is *entailed* by $\mathcal{K}$ (at time point i) if the set $\mathsf{Cert}(\Phi, \mathcal{K})$ ($\mathsf{Cert}(\Phi, \mathcal{K}, i)$) is non-empty. In this case, we write $\mathcal{K} \models \Phi$ ($\mathcal{K}, i \models \Phi$), and similarly for $\mathfrak{I} \models \Phi$ and $\mathfrak{I}, i \models \Phi$.

As in the previous chapters, we here apply the standard semantics (see Section 3.1). Observe, however, that this may have some unintended consequences in practice. For example, the PTQ $\diamond_P \Box_F \texttt{Running}(\texttt{process1})$ then always is false, although, at the current time point, the process may have been running since some past time point, and the future state of the system is unknown.

An alternative semantics, which overcomes this problem, does not consider time points before 0 or after n; such a temporal semantics is, for example, used for LTL in [Wil99] or for temporal query languages for databases [SL89; HS91b; Cho95] and might also be reasonable for applications of temporal query answering. In fact, we consider this semantics in [BLT15], which means that our rewritability result, presented in the next section, extends to that setting. This alternative semantics has the effect that the PTQ $\bigcirc_F \mathit{true}$ is not entailed at the last time point; the semantics of the $\bigcirc_F$-operator thus differs from that given in Figure 8.3: $\mathfrak{I}, i \models \mathfrak{a}(\bigcirc_F \Phi_1)$ iff $i < n$ and $\mathfrak{I}, i+1 \models \mathfrak{a}(\Phi_1)$. At first glance, this may seem counterintuitive, but it is often considered to be more natural to restrict the aggregation operators to the time points for which data is available. For instance, current systems for stream reasoning restrict the focus to a so-called *window* of the data stream (see Section 1.3 for an overview of current stream reasoning approaches).

A compromise between those two kinds of semantics could be obtained by "looping" the last interpretation or fact base infinitely often, meaning that the facts of the last time point stay valid forever. This would make the PTQ $\bigcirc_F \mathit{true}$ equivalent to *true*,

while retaining the spirit of the finite semantics. However, also this semantics has counterintuitive side-effects, as it makes severe assumptions on the future. For example, then, the PTQ $\Box_F$`Running(process1)` is true at n if (and only if) `Running(process1)` is true at n.

8.3 A Rewritability Result

Targeting temporal query answering, we finally lift the rewriting approach introduced in Section 8.1.3 to the temporal setting. We show that, under the assumptions formulated in the previous sections, *positive temporal* $\mathcal{QL}_1$ queries also enjoy a rewritability property—w.r.t. temporal knowledge bases formulated in $\mathcal{L}$—and how the certain answers to positive temporal $\mathcal{QL}_1$ queries over $\mathcal{L}$ can be computed. Recall our assumption on the properties of the query languages $\mathcal{QL}_1$ and $\mathcal{QL}_2$ and logic $\mathcal{L}$:

(P1) Consistency of knowledge bases in $\mathcal{L}$ should be decidable. This is a basic prerequisite for any reasoning procedure, in particular for query answering.

(P2) The logic $\mathcal{L}$ should have the canonical model property w.r.t. $\mathcal{QL}_1$ (see Definition 8.9). This property often represents a first step towards a rewritability result and is similarly important for our result.

(P3) $\mathcal{QL}_1$ queries should be $\mathcal{QL}_2$ rewritable w.r.t. $\mathcal{L}$. In particular, we make use of $\Delta_\mathcal{T}$, $\mathcal{D}_\mathcal{K}$, and $\varphi^\mathcal{T}$ introduced in Definition 8.11.

(P4) The set of answers to any $\mathcal{QL}_2$ query w.r.t. a finite interpretation should be computable.

Observe that we do not need (P4) in order to obtain our result, but the latter would otherwise be useless.

Before providing the rewritability result, we first lift the constructions of Definitions 8.9 and 8.11 to the temporal setting. For this, consider a temporal $\mathcal{QL}_1$ query Φ and a consistent TKB $\mathcal{K} = \langle \mathcal{T}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$, which is possible by (P1); to see the latter, observe that the consistency of the TKB can be decided by regarding the KBs $\langle \mathcal{T}, \mathcal{A}_i \rangle$ with $i \in [0, n]$ separately. That is, we can also assume these atemporal KBs to be consistent, and thus define the sequences $\mathfrak{I}_\mathcal{K} := (\mathcal{I}_{\mathcal{K}_i})_{i \geq 0}$ and $\mathfrak{D}_\mathcal{K} := (\mathcal{D}_{\mathcal{K}_i})_{i \geq 0}$ of canonical and finite interpretations by (P2) and (P3); for all $i > n$, we consider the canonical and finite interpretations of $\langle \mathcal{T}, \emptyset \rangle$. Due to the assumption that each $\mathcal{I}_{\mathcal{K}_i}$ is countably infinite and Condition (ii) of Definition 8.6, we can without loss of generality assume that these canonical models have the same domain. Similarly, the finite interpretations $\mathcal{D}_{\mathcal{K}_i}$ have the common domain $\Delta_\mathcal{T}$. Thus, they are valid sequences of interpretations according to our semantics. We then define the temporal $\mathcal{QL}_2$ query $\Phi^\mathcal{T}$ as the one obtained from Φ by replacing every $\mathcal{QL}_1$ query φ occurring in Φ by the $\mathcal{QL}_2$ query $\varphi^\mathcal{T}$ and get the following rewritability result.

Theorem 8.15 *Let $\mathcal{QL}_1$ and $\mathcal{QL}_2$ be query languages and $\mathcal{L}$ be a logic that has the canonical model property w.r.t. $\mathcal{QL}_1$, such that $\mathcal{QL}_1$ queries are $\mathcal{QL}_2$ rewritable w.r.t. $\mathcal{L}$.*

Then, for every consistent TKB $\mathcal{K} = \langle \mathcal{T}, (\mathcal{A}_i)_{0 \leq i \leq n} \rangle$*, every temporal* $\mathcal{QL}_1$ *query* Φ*, and every* $i \in [0, n]$*, we have:*

$$\mathsf{Cert}(\Phi, \mathcal{K}, i) = \mathsf{Ans}(\Phi, \mathfrak{I}_\mathcal{K}, i) = \mathsf{Ans}(\Phi^\mathcal{T}, \mathfrak{D}_\mathcal{K}, i).$$

Proof. We first prove $\mathsf{Cert}(\Phi, \mathcal{K}, i) \subseteq \mathsf{Ans}(\Phi, \mathfrak{I}_\mathcal{K}, i)$. For some $\mathfrak{a} \in \mathsf{Cert}(\Phi, \mathcal{K}, i)$ and every $\mathfrak{I} = (\mathcal{I}_i)_{0 \leq i \leq n}$ with $\mathfrak{I} \models \mathcal{K}$, we then have $\mathfrak{I}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi)$. By (P2) and Definition 8.14, we get $\mathfrak{I}_\mathcal{K}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi)$ or, equivalently, $\mathfrak{a} \in \mathsf{Ans}(\Phi, \mathfrak{I}_\mathcal{K}, i)$.

It is left to prove the following two claims:

(1) $\mathsf{Ans}(\Phi, \mathfrak{I}_\mathcal{K}, i) \subseteq \mathsf{Ans}(\Phi^\mathcal{T}, \mathfrak{D}_\mathcal{K}, i)$,

(2) $\mathsf{Ans}(\Phi^\mathcal{T}, \mathfrak{D}_\mathcal{K}, i) \subseteq \mathsf{Cert}(\Phi, \mathcal{K}, i)$.

We show this by induction on the structure of Φ.

For the base case, we regard an atemporal $\mathcal{QL}_1$ query Φ. For (1), let $\mathfrak{a} \in \mathsf{Ans}(\Phi, \mathfrak{I}_\mathcal{K}, i)$. Since Φ is a $\mathcal{QL}_1$ query, the semantics yields that $\mathfrak{a} \in \mathsf{Ans}(\Phi, \mathcal{I}_{\mathcal{K}_i})$. From (P3), we obtain $\mathfrak{a} \in \mathsf{Ans}(\Phi^\mathcal{T}, \mathcal{D}_{\mathcal{K}_i})$, and thus $\mathfrak{a} \in \mathsf{Ans}(\Phi^\mathcal{T}, \mathfrak{D}_\mathcal{K}, i)$, by the semantics of temporal $\mathcal{QL}_2$ queries.

For (2), let $\mathfrak{a} \in \mathsf{Ans}(\Phi^\mathcal{T}, \mathfrak{D}_\mathcal{K}, i)$. Since $\Phi^\mathcal{T}$ is a $\mathcal{QL}_2$ query, this implies $\mathfrak{a} \in \mathsf{Ans}(\Phi^\mathcal{T}, \mathcal{D}_{\mathcal{K}_i})$. Because of (P3), we have $\mathfrak{a} \in \mathsf{Cert}(\Phi, \mathcal{K}_i)$. This means that for every interpretation $\mathcal{I}$ with $\mathcal{I} \models \mathcal{A}_i$ and $\mathcal{I} \models \mathcal{T}$, we have that $\mathcal{I} \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi)$. Hence, for every sequence $\mathfrak{I} = (\mathcal{I}_i)_{0 \leq i \leq n}$ with $\mathfrak{I} \models \mathcal{K}$, we have $\mathcal{I}_i \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi)$. Since Φ is a $\mathcal{QL}_1$ query, the latter condition is equivalent to $\mathfrak{a} \in \mathsf{Ans}(\Phi, \mathfrak{I}, i)$, and thus we get $\mathfrak{a} \in \mathsf{Cert}(\Phi, \mathcal{K}, i)$.

Let now Φ be of the form $\Phi_1 \wedge \Phi_2$. For (1), assume that $\mathfrak{I}_\mathcal{K}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi)$, and thus we have $\mathfrak{I}_\mathcal{K}, i \models_{\mathcal{QL}_1} \mathfrak{a}_{\Phi_1}(\Phi_1)$ and $\mathfrak{I}_\mathcal{K}, i \models_{\mathcal{QL}_1} \mathfrak{a}_{\Phi_2}(\Phi_2)$. By the induction hypothesis, $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_1}(\Phi_1^\mathcal{T})$ and $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_2}(\Phi_2^\mathcal{T})$, and thus we get $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi^\mathcal{T})$ by the definition of $\Phi^\mathcal{T}$ and the semantics.

For (2), we assume that $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi^\mathcal{T})$, and thus $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_1}(\Phi_1^\mathcal{T})$ and $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_2}(\Phi_2^\mathcal{T})$. Hence, we have $\mathfrak{a} \in \mathsf{Cert}(\Phi_1, \mathcal{K}, i)$ and $\mathfrak{a} \in \mathsf{Cert}(\Phi_2, \mathcal{K}, i)$ by the induction hypothesis. Thus, for every $\mathfrak{I}$ with $\mathfrak{I} \models \mathcal{K}$, it holds that $\mathfrak{I}, i \models_{\mathcal{QL}_1} \mathfrak{a}_{\Phi_1}(\Phi_1)$ and $\mathfrak{I}, i \models_{\mathcal{QL}_1} \mathfrak{a}_{\Phi_2}(\Phi_2)$. This is equivalent to $\mathfrak{a} \in \mathsf{Cert}(\Phi_1 \wedge \Phi_2, \mathcal{K}, i)$.

Let now Φ be of the form $\bigcirc_F \Phi_1$. For Claim (1), we take $\mathfrak{I}_\mathcal{K}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\bigcirc_F \Phi_1)$. By the temporal semantics, we have $\mathfrak{I}_\mathcal{K}, i+1 \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi_1)$. By the induction hypothesis, we get $\mathfrak{D}_\mathcal{K}, i+1 \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi_1^\mathcal{T})$, which implies $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi^\mathcal{T})$ by the definition of $\Phi^\mathcal{T}$.

For (2), let $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi^\mathcal{T})$. Hence, we have $\mathfrak{D}_\mathcal{K}, i+1 \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi_1^\mathcal{T})$, which implies $\mathfrak{a} \in \mathsf{Cert}(\Phi_1, \mathcal{K}, i+1)$ by the induction hypothesis. This means that, for every $\mathfrak{I} \models \mathcal{K}$, we have $\mathfrak{I}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\bigcirc_F \Phi_1)$, which shows that $\mathfrak{a} \in \mathsf{Cert}(\Phi, \mathcal{K}, i)$.

For the next inductive case, let Φ be of the form $\Phi_1 \,\mathcal{U}\, \Phi_2$. For (1), we assume that $\mathfrak{I}_\mathcal{K}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi_1 \,\mathcal{U}\, \Phi_2)$, and thus there is a $k \geq i$ such that $\mathfrak{I}_\mathcal{K}, k \models_{\mathcal{QL}_1} \mathfrak{a}_{\Phi_2}(\Phi_2)$ and $\mathfrak{I}_\mathcal{K}, j \models_{\mathcal{QL}_1} \mathfrak{a}_{\Phi_1}(\Phi_1)$ for all $j \in [i, k[$. By the induction hypothesis, we obtain $\mathfrak{D}_\mathcal{K}, k \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_2}(\Phi_2^\mathcal{T})$ and $\mathfrak{D}_\mathcal{K}, j \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_1}(\Phi_1^\mathcal{T})$ for all $j \in [i, k[$. The definitions of $\models_{\mathcal{QL}_2}$ and $\Phi^\mathcal{T}$ yield that $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi^\mathcal{T})$.

For (2), we assume that $\mathfrak{D}_\mathcal{K}, i \models_{\mathcal{QL}_2} \mathfrak{a}(\Phi^\mathcal{T})$. By the definition of $\Phi^\mathcal{T}$, there is a $k \geq i$ with $\mathfrak{D}_\mathcal{K}, k \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_2}(\Phi_2^\mathcal{T})$ and $\mathfrak{D}_\mathcal{K}, j \models_{\mathcal{QL}_2} \mathfrak{a}_{\Phi_1}(\Phi_1^\mathcal{T})$ for all $j \in [i, k[$. The induction hypothesis yields $\mathfrak{a} \in \mathsf{Cert}(\Phi_2, \mathcal{K}, k)$ and $\mathfrak{a} \in \mathsf{Cert}(\Phi_1, \mathcal{K}, j)$ for all $j \in [i, k[$. As a consequence, we have for every $\mathfrak{I} \models \mathcal{K}$ that $\mathfrak{I}, i \models_{\mathcal{QL}_1} \mathfrak{a}(\Phi_1 \,\mathcal{U}\, \Phi_2)$.

The remaining cases can be proven in a similar way. For example, the arguments for $\bigcirc_P \Phi_1$ can be obtained from those of case $\bigcirc_F \Phi_1$ by replacing $i+1$ by $i-1$, and correspondingly for $\Phi_1 \mathcal{S} \Phi_2$ and $\Phi_1 \mathcal{U} \Phi_2$. The cases for $\Box$ and $\Box_P$ follow from similar arguments. □

8.4 Summary

In this chapter, we have generalized the abstract setting for ontology-based temporal query answering introduced in Chapter 3 even more. In particular, we have shown that the temporal query answering problem is rewritable in certain cases if the negation in the queries is dropped. This generality allows to augment many existing query answering approaches with temporal features: both the temporal queries and the ontologies can be instantiated with arbitrary queries and logical theories, as long as they satisfy certain requirements. And we have shown that many formalisms already applied or proposed in the literature do so.

The rewritability result is still of theoretical nature, but we have proposed algorithms for answering PTQs in [BLT15].[8] The focus of the latter work is on the *temporal database monitoring problem*: the continuous evaluation of a fix set of PTQs over a temporal knowledge base, which contains data about the past (and present), growing over time. We have identified three different approaches for solving that problem.

The most straightforward option is to evaluate PTQs in a database system that supports temporal information or in a data stream processing system. The advantage of this approach is that the optimization techniques of the database can directly be exploited. Yet, it requires to store the whole history of past data—even if only a small part of it is necessary to answer the query—and to re-evaluate the query at each time point using a temporal database query language, such as ATSQL [CTB01]. The feasibility thus depends on the amount of data that has to be considered. Note that many existing stream processing systems limit the latter by adopting a "sliding view" semantics, in which only a fixed amount of past time points is used to evaluate PTQs.

Second, we can apply an approach proposed by [Cho95; Tom04], which achieves a so-called *bounded history encoding*; that is, the amount of data that is required to answer the given queries is bounded. In the algorithm, it is continuously updated as soon as new data is available. However, since the original proposal disregards future operators, they have to be eliminated in an extra step; in [BLT15], we describe how this can be done. Although this elimination is independent of the length of the history, it involves a theoretical non-elementary blowup in the size of the query, due to the application of the separation theorem (see Lemma 2.20). An advantage of the history encoding from [Cho95; Tom04] is that it can be implemented inside a database system using views and triggers, which could yield a good performance in spite of the possibly very large size of the query. Generally, this option is the best of the three if the PTQs contain no future operators or if one can find a small equivalent representation without future operators.

As most general solution, we propose a new algorithm, which is an adaptation of the one proposed in [Cho95]. Our algorithm allows for rigid unary predicates (see

[8] Recall that we consider a slightly different semantics in [BLT15].

Section 3.1), directly works with future operators, and also achieves a bounded history encoding. In particular, we can limit the influence of the future operators on the time and space requirements to a single exponential factor. But it is not straightforward how this algorithm can be implemented inside a database system. While it is theoretically the most efficient solution, it remains to be seen how it implementations perform in practice.

9 Conclusions

The goal of this thesis was to systematically analyze ontology-based access to temporal data in terms of computational complexity, and rewritability to existing formalisms. In this chapter, we summarize our achievements and describe directions of future research.

9.1 Summary of Results

In this work, we have focused on a temporal query answering scenario that reflects the needs of the applications of today: the temporal queries are based on LTL, one of the most important temporal logics; the ontologies are written in standard lightweight logics; and the data allows to capture data streams.

In Chapter 3, we have introduced an abstract temporal query language that combines atemporal queries $\mathcal{QL}$ via the operators of linear temporal logic and allows to access temporal data through ontologies. In particular, it generalizes existing temporal query languages and, at the same time, provides a framework for the design of new formalisms and general investigations. We have proven that its direct application yields only containment in NExpTime (NP) w.r.t. combined (data) complexity,[1] which does not fit the low complexity we usually have with lightweight DLs; regarding entailment, we thus get containment in co-NExpTime (co-NP). For that reason, we have proposed a new approach based on the original algorithm for solving the satisfiability problem in LTL, which requires only polynomial space. This approach is similarly general w.r.t. the query language and DL considered, but leaves one part of the TQ satisfiability problem, r-satisfiability, open and thus to be solved for concrete TQs and DLs.

In Chapter 4, we have studied the combined complexity of the satisfiability problem in $\mathcal{DL}$-LTL for DLs $\mathcal{DL}$ meeting a few rather weak requirements. Moreover, we have showed that the latter are satisfied in the popular DL $\mathcal{EL}$ and many *DL-Lite* fragments. Nevertheless, $\mathcal{DL}$-LTL has turned out to be powerful enough to show a lower bound of NExpTime if rigid symbols are considered. For the case without rigid symbols, we have shown containment in PSpace, based on our new approach. For $\mathcal{EL}$ and $\mathit{DL\text{-}Lite}^{\mathcal{H}}_{horn}$, this PSpace result also holds w.r.t. rigid symbols if the considered concept inclusions are global. An overview of these results is given in Figure 4.1.

In Chapter 5, we have focused on TCQ entailment in $\mathcal{EL}$ and showed results similar to those for $\mathcal{EL}$-LTL w.r.t. combined complexity (see Figure 9.1): our general approach can be applied to design a polynomial-space algorithm, and co-NExpTime-hardness can be shown similarly. However, in contrast to $\mathcal{EL}$-LTL, the former also holds for the case with rigid concept names. This shows that the local GCIs allowed in $\mathcal{EL}$-LTL are rather powerful; note that $\mathcal{EL}$-LTL formulas without them can be seen as TCQs w.r.t. a

[1] Recall that, in order to obtain this complexity, the satisfiability of conjunctions of $\mathcal{QL}$ queries and negated $\mathcal{QL}$ queries w.r.t. a KB in $\mathcal{DL}$ has to be decidable nondeterministically in polynomial time.

	Data Complexity			Combined Complexity		
	(i)	(ii)	(iii)	(i)	(ii)	(iii)
$\textit{DL-Lite}^{[\ \mid\mathcal{H}]}_{[core\mid horn]}$	ALogTime	ALogTime	ALogTime	PSpace	PSpace	PSpace
	≥ Th. 6.31		≤ Th. 6.37	≥ [SC85]		≤ Co. 6.19
$\mathcal{EL}$	P	co-NP	co-NP	PSpace	PSpace	co-NExpTime
	≥ [Cal+06], ≤ Th. 5.19	≥ Th. 5.21	≤ Co. 5.20	≥ [SC85]	≤ Co. 5.16	≥ Th. 5.18, ≤ Co. 5.17
$\mathcal{ALC}$-$\mathcal{SHQ}$[a]	co-NP	co-NP	≤ExpTime	ExpTime	co-NExpTime	2-ExpTime
$\textit{DL-Lite}_{[krom\mid bool]}$	co-NP	co-NP	≤ExpTime	ExpTime	co-NExpTime	2-ExpTime
	≥ [Cal+05]			≥ Co. 7.5, ≤ Th. 7.6	≥ Th. 7.7, ≤ Th. 7.8	≥ Th. 7.9
$\textit{DL-Lite}^{\mathcal{H}}_{[krom\mid bool]}$	co-NP	co-NP	≤ExpTime	2-ExpTime	2-ExpTime	2-ExpTime
		≤ Co. 7.12	≤ Co. 7.12	≥ Co. 7.5		≤ [BBL15a]

Figure 9.1: The complexity of TCQ entailment considering (i) no rigid symbols, (ii) rigid concept names, and (iii) rigid role names. Our results are highlighted. All complexities except those marked with ≤ are tight; ≥ hardness, ≤ containment.

[a] [BBL15b]

global ontology. Regarding data complexity, tractability only holds for the case without rigid symbols.

In Chapter 6, we have focused on TCQ entailment in Horn fragments of *DL-Lite*, including role inclusions, which offer expressive features that are rather similar to those of $\mathcal{EL}$. But the results we have shown are considerably better: containment in PSpace and ALogTime w.r.t. combined and data complexity. Although we could not achieve FO rewritability, the latter result is interesting since containment in ALogTime is considered as an indicator for the existence of efficient parallel implementations [AB09, Thm. 6.27]; also recall that, in many applications, data complexity better captures resource consumption than combined complexity (see Section 2.3). To achieve both of these results, we have shown that polynomially many stored assertions and queries, if guessed before processing, suffice to test r-satisfiability; that is, to testy¿y if the data of one observation moment does not contradict rigid knowledge about past moments.

In Chapter 7, we have considered TCQ entailment regarding the two DLs $\textit{DL-Lite}_{krom}$ and $\textit{DL-Lite}_{bool}$, also with role inclusions, which are no Horn logics. These DLs are rather expressive, but do not allow to model qualified existential restrictions on the left-hand side of concept inclusions. As described above, we have identified this feature as a cause of complexity for TCQ answering; recall that it is the main difference between $\mathcal{EL}$ and $\textit{DL-Lite}_{horn}$, which instead allows for inverse roles. For that reason, the results of Chapter 7 are especially interesting. We have shown that entailment in $\textit{DL-Lite}_{bool}$ can be reduced to entailment in $\textit{DL-Lite}_{krom}$. TCQ entailment in these logics generally has turned out to be as complex as in more expressive DLs, such as $\mathcal{SHQ}$ [BBL15b]. Further, we have shown that role inclusions, which allow to express qualified existential restrictions on the right-hand side of concept inclusions, lead to 2-ExpTime-completeness in combined complexity, which is even higher than the results proven for very expressive DLs [BBL15b]. Note that further results on TCQ entailment in DLs extending $\mathcal{SHQ}$ are presented in [BBL15a]. An interesting observation regarding those DLs is that the complexity of CQ entailment is often already higher than the one of TCQ entailment

in $\mathcal{SHQ}$. Nevertheless, the complexities for several cases are still open, also for CQ entailment.

In Chapter 8, we have investigated query answering, considered temporal query languages based on many different atemporal query languages studied in the literature, and regarded various different lightweight logics as ontology languages. In particular, we have achieved a generic rewritability result by disallowing negation in the temporal queries.

Altogether, our results show that the features we have studied can often be considered "for free":

- Regarding combined complexity, we have shown that there are many popular DLs for which the problems of $\mathcal{DL}$-LTL satisfiability or TCQ entailment w.r.t. a TKB are in PSPACE, even if rigid symbols are considered. This matches the complexity of satisfiability in LTL, which is much less expressive given the fact that ontologies are not considered at all.

- Comparing the data complexity of TCQ entailment to that of CQ entailment, we only get similar results for $\mathcal{EL}$ in the case without rigid symbols, and for the expressive *DL-Lite* logics in the cases without rigid roles[2]; but the TCQs still provide much more expressivity. Moreover, the results for $\textit{DL-Lite}_{horn}^{\mathcal{H}}$ are not much higher than the AC^0-containment given for standard CQ entailment and even hold w.r.t. rigid symbols.

- There exist many rewritable query languages $\mathcal{QL}$ and lightweight logics $\mathcal{DL}$ that satisfy the requirements which we show to imply that positive temporal $\mathcal{QL}$ queries w.r.t. $\mathcal{DL}$ TKBs are similarly rewritable.

9.2 Future Work

Our results can be extended in two directions: on the theoretical side, it is left to close some gaps and to push the envelope of the promising results by considering more expressive query and ontology languages; on the other hand, many questions are left open regarding the practical impact of our results.

In this work, we have focused on the standard LTL operators in queries and basic lightweight DLs as ontology languages. We consider several follow-up questions to be especially interesting:

- What *tight data complexity results* do we get for TCQ entailment in the more expressive *DL-Lite* fragments if rigid role names are considered? The difference between in CO-NP and in EXPTIME should have an impact on practical considerations.

- For which *extensions of $\mathcal{EL}$ and DL-Lite* do we get similar complexity results? This may be particularly interesting regarding $\mathcal{ELO}_\bot(\mathcal{D})$[3], the DL closest to OWL 2 EL for which we may get such results—in $\mathcal{EL}^+$ and $\mathcal{EL}^{++}$, CQ answering is

[2] Note that the result for the case with rigid roles is not tight.

[3] $\mathcal{ELO}_\bot(\mathcal{D})$ extends $\mathcal{ELO}_\bot$ in that it allows to use concrete domains.

undecidable [Ros07, Thm. 3]. Recall that *DL-Lite*$_R$, the DL closest to OWL 2 QL extends *DL-Lite*$_{core}$ with role inclusions and allows to express disjointness of roles. Since we consider the former with *DL-Lite* and the latter can be easily modeled with TCQs, our results for *DL-Lite*$_{core}^{\mathcal{H}}$ extend to *DL-Lite*$_R$. Yet, there are many other features that can be regarded with *DL-Lite* and which we have not considered; for example, functionality or number restrictions [Art+09]. We also estimate the possibility to refer to concrete values in the ontology to be particularly relevant in practice. Hence, another useful extension for both the query and the ontology language could be concrete domains [BH91].

- Are there *restrictions* that yield better complexities and constraints different from the ones we have proposed for achieving rewritability? Since several restrictions have turned out to be effective for temporal lightweight DLs [Art+15a; GJK16], also our results could be augmented by considering, for instance, acyclicity conditions for the ontology.

- What is the influence of *metric temporal operators* in the queries? First steps in the direction of metric temporal DLs have recently been made [GJO16; Baa+17]. Since there are metric extensions of LTL for which the complexity does not differ from that of LTL [LWW07], it would be interesting to see if our approaches are robust enough or can be extended to work for temporal queries with such operators.

- Can we also achieve good results if we allow some *temporal operators in the ontology*? That is, finally, our work could be integrated with recent advances on temporal lightweight DLs [Art+15a; GJK16], which show that rewritability can be obtained sometimes and that there are several kinds of restriction that allow for tractable query answering.

On the practical side, several questions arise directly:

- Are there efficient *algorithms for solving the satisfiability problem* for temporal queries? Our PSpace results rely on the LTL satisfiability algorithm proposed in [SC85]. Practical algorithms solving LTL satisfiability are however usually exponential-time approaches or apply special techniques, such as parallelization, to cope with the nondeterminism since the guessing employed in the original algorithm is hardly feasible in applications [Wul+08; Li+13; Li+15]. It is still open which kinds of algorithms are useful in applications of ontology-based temporal query answering, in particular, if rigid symbols are considered. On the other hand, application knowledge discerning rigid symbols in advance could improve performance.

- For the cases where we have shown promising results for query entailment, do there exist *efficient algorithms for query answering*? In general, algorithms solving query entailment do not allow for efficient query answering since the latter problem, in practice, is harder to solve (see Section 3.1).

- Can the *rewritability result easily be implemented* based on existing systems rewriting atemporal queries? Recall that it has been rather easily obtained from existing

rewritability results for the atemporal case. In [BLT15], we describe different algorithms for temporal query answering that rely on such existing approaches, but the implementation is still future work.

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[AF00] Alessandro Artale and Enrico Franconi. "A Survey of Temporal Extensions of Description Logics". In: *Annals of Mathematics and Artificial Intelligence* 30.1-4 (2000), pp. 171–210.

[AF05] Alessandro Artale and Enrico Franconi. "Temporal Description Logics". In: *Handbook of Temporal Reasoning in Artificial Intelligence*. Ed. by Michael Fisher, Dov M. Gabbay, and Lluis Vila. Elsevier Science Inc., 2005, pp. 375–388.

[AF98] Alessandro Artale and Enrico Franconi. "A Temporal Description Logic for Reasoning about Actions and Plans". In: *Journal of Artificial Intelligence Research* 9 (1998), pp. 463–506.

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison Wesley, 1995.

[Ani+12] Darko Anicic, Sebastian Rudolph, Paul Fodor, and Nenad Stojanovic. "Stream Reasoning and Complex Event Processing in ETALIS". In: *Semantic Web Journal* 3.4 (2012), pp. 397–407.

[Ara+08] José Arancón, Luis Polo, Diego Berrueta, François-Marie Lesaffre, Nicolàs Abajo, and Antonio M. Campos. "Ontology-Based Knowledge Management In The Steel Industry". In: *The Semantic Web: Real-World Applications from Industry*. Ed. by Jorge Cardoso, Martin Hepp, and Miltiadis D. Lytras. Springer-Verlag, 2008, pp. 243–272.

[Art+07] Alessandro Artale, Roman Kontchakov, Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. "Temporalising Tractable Description Logics". In: *Proc. of the 14th Int. Symposium on Temporal Representation and Reasoning (TIME'07)*. Ed. by Valentin Goranko and X. Sean Wang. IEEE Press, 2007, pp. 11–22.

[Art+09] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. "The *DL-Lite* Family and Relations". In: *Journal of Artificial Intelligence Research* 36 (2009), pp. 1–69.

[Art+14a] Alessandro Artale, Davide Bresolin, Angelo Montari, Guido Sciavicco, and Vladislav Ryzhikov. "*DL-Lite* and Interval Temporal Logics: A Marriage Proposal". In: *Proc. of the 21st European Conference on Artificial Intelligence (ECAI'14)*. Ed. by Torsten Schaub. Vol. 263. Frontiers in Artificial Intelligence and Applications. IOS Press, 2014, pp. 957–958.

[Art+14b] Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. "A Cookbook for Temporal Conceptual Data Modelling with Description Logics". In: *ACM Transactions on Computational Logic* 15.3 (2014), p. 25.

[Art+15a] Alessandro Artale, Roman Kontchakov, Alisa Kovtunova, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. "First-Order Rewritability of Ontology-Mediated Temporal Queries". In: *Proc. of the 24th Int. Joint Conference on Artificial Intelligence (IJCAI'15)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 2706–2712.

[Art+15b] Alessandro Artale, Roman Kontchakov, Vladislav Ryzhikov, and Michael Zakharyaschev. "Tractable Interval Temporal Propositional and Description Logics". In: *Proc. of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. Ed. by Blai Bonet and Sven Koenig. AAAI Press, 2015, pp. 1417–1423.

[Baa+07] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, eds. *The Description Logic Handbook: Theory, Implementation, and Applications*. 2nd ed. Cambridge University Press, 2007.

[Baa+17] Franz Baader, Stefan Borgwardt, Patrick Koopmann, Ana Ozaki, and Veronika Thost. "Metric Temporal Description Logics with Interval-Rigid Names". In: *Proc. of the 11th Int. Symposium on Frontiers of Combining Systems (FroCoS'17)*. Vol. 10483. Lecture Notes in Computer Science. To appear. Springer-Verlag, 2017.

[Baa03] Franz Baader. "Terminological Cycles in a Description Logic with Existential Restrictions". In: *Proc. of the 18th Int. Joint Conference on Artificial Intelligence (IJCAI'03)*. Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, 2003, pp. 325–330.

[BAC10] Elena Botoeva, Alessandro Artale, and Diego Calvanese. "Query Rewriting in $\mathit{DL\text{-}Lite}_{horn}^{(\mathcal{HN})}$". In: *Proc. of the 2010 Int. Workshop on Description Logics (DL'10)*. Ed. by Volker Haarslev, David Toman, and Grant Weddell. Vol. 573. CEUR Workshop Proceedings. 2010, pp. 267–278.

[Bag+11a] Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat. "On Rules with Existential Variables: Walking the Decidability Line". In: *Artificial Intelligence* 175.9–10 (2011), pp. 1620–1654.

[Bag+11b] Jean-François Baget, Marie-Laure Mugnier, Sebastian Rudolph, and Michaël Thomazo. "Walking the Complexity Lines for Generalized Guarded Existential Rules". In: *Proc. of the 22nd Int. Joint Conference on Artificial Intelligence (IJCAI'11)*. Ed. by Toby Walsh. AAAI Press, 2011, pp. 712–717.

[BBL05] Franz Baader, Sebastian Brandt, and Carsten Lutz. "Pushing the $\mathcal{EL}$ Envelope". In: *Proc. of the 19th Int. Joint Conference on Artificial Intelligence (IJCAI'05)*. Ed. by Leslie Pack Kaelbling and Alessandro Saffiotti. Professional Book Center, 2005, pp. 364–369.

[BBL13] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. "Temporalizing Ontology-Based Data Access". In: *Proc. of the 24th Int. Conference on Automated Deduction (CADE'13)*. Ed. by Maria Paola Bonacina. Vol. 7898. Lecture Notes in Computer Science. Springer-Verlag, 2013, pp. 330–344.

[BBL15a] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. "Temporal Conjunctive Queries in Expressive Description Logics with Transitive Roles". In: *Proc. of the 28th Australasian Joint Conference on Artificial Intelligence (AI'15)*. Ed. by Bernhard Pfahringer and Jochen Renz. Vol. 9457. Lecture Notes in Artificial Intelligence. Springer-Verlag, 2015, pp. 21–33.

[BBL15b] Franz Baader, Stefan Borgwardt, and Marcel Lippmann. "Temporal Query Entailment in the Description Logic $\mathcal{SHQ}$". In: *Journal of Web Semantics* 33 (2015), pp. 71–93.

[BCW93] Marianne Baudinet, Jan Chomicki, and Pierre Wolper. "Temporal Deductive Databases". In: (1993). Ed. by Abdullah Uz Tansel, James Clifford, Shashi Gadia, Sushil Jajodia, Arie Segev, and Richard Snodgrass, pp. 294–320.

[Bec+15] Harald Beck, Minh Dao-Tran, Thomas Eiter, and Michael Fink. "LARS: A Logic-based Framework for Analyzing Reasoning over Streams". In: *Proc. of the 29th AAAI Conference on Artificial Intelligence*. Ed. by Blai Bonet and Sven Koenig. AAAI Press, 2015, pp. 1431–1438.

[BGG97] Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Perspectives in Mathematical Logic. Springer-Verlag, 1997.

[BGL12] Franz Baader, Silvio Ghilardi, and Carsten Lutz. "LTL over Description Logic Axioms". In: *ACM Transactions on Computational Logic* 13.3 (2012), 21:1–21:32.

[BH91] Franz Baader and Philipp Hanschke. "A Scheme for Integrating Concrete Domains into Concept Languages". In: *Proc. of the 12th Int. Joint Conference on Artificial Intelligence (IJCAI'91)*. Ed. by John Mylopoulos and Raymond Reiter. Vol. 1. Morgan Kaufmann, 1991, pp. 446–451.

[BHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. "The Semantic Web". In: *Scientific American* (2001), pp. 96–101.

[Bie+14] Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter. "Ontology-Based Data Access: A Study through Disjunctive Datalog, CSP, and MMSNP". In: *ACM Transactions on Database Systems* 39.4 (2014), 33:1–33:44.

[Bie+16] Meghyn Bienvenu, Peter Hansen, Carsten Lutz, and Frank Wolter. "First Order-Rewritability and Containment of Conjunctive Queries in Horn Description Logics". In: *Proc. of the 25th Int. Joint Conference on Artificial Intelligence (IJCAI'16)*. Ed. by Subbarao Kambhampati. AAAI Press, 2016, pp. 965–971.

[BIS90] David A. M. Barrington, Neil Immerman, and Howard Straubing. "On Uniformity Within NC^1". In: *Journal of Computer and System Sciences* 41.3 (1990), pp. 274–306.

[BKM99] Franz Baader, Ralf Küsters, and Ralf Molitor. "Computing Least Common Subsumers in Description Logics with Existential Restrictions". In: *Proc. of the 16th Int. Joint Conference on Artificial Intelligence (IJCAI'99)*. Ed. by Thomas Dean. Vol. 1. Morgan Kaufmann, 1999, pp. 96–101.

[BLT13a] Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. "Temporal Query Answering in *DL-Lite*". In: *Proc. of the 26th Int. Workshop on Description Logics (DL'13)*. Ed. by Thomas Eiter, Birte Glimm, Yevgeny Kazakov, and Markus Krötzsch. Vol. 1014. CEUR Workshop Proceedings. 2013, pp. 80–92.

[BLT13b] Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. "Temporal Query Answering in the Description Logic *DL-Lite*". In: *Proc. of the 9th Int. Symposium on Frontiers of Combining Systems (FroCoS'13)*. Ed. by Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt. Vol. 8152. Lecture Notes in Computer Science. Springer-Verlag, 2013, pp. 165–180.

[BLT15] Stefan Borgwardt, Marcel Lippmann, and Veronika Thost. "Temporalizing Rewritable Query Languages over Knowledge Bases". In: *Journal of Web Semantics* 33 (2015), pp. 50–70.

[BMP13] Pierre Bourhis, Michael Morak, and Andreas Pieris. "The Impact of Disjunction on Query Answering Under Guarded-Based Existential Rules". In: *Proc. of the 23rd Int. Joint Conference on Artificial Intelligence (IJCAI'13)*. Ed. by Francesca Rossi. AAAI Press, 2013, pp. 796–802.

[BOS15] Meghyn Bienvenu, Magdalena Ortiz, and Mantas Simkus. "Regular Path Queries in Lightweight Description Logics: Complexity and Algorithms". In: *Journal of Artificial Intelligence Research* 53 (2015), pp. 315–374.

[Bou+16] Pierre Bourhis, Marco Manna, Michael Morak, and Andreas Pieris. "Guarded-Based Disjunctive Tuple-Generating Dependencies". In: *ACM Transactions on Database Systems* 41.4 (2016), 27:1–27:45.

[Bra04] Sebastian Brandt. "Polynomial Time Reasoning in a Description Logic with Existential Restrictions, GCI Axioms, and—What Else?" In: *Proc. of the 16th European Conference on Artificial Intelligence (ECAI'04)*. Ed. by R. López de Mantáras and L. Saitta. IOS Press, 2004, pp. 298–302.

[BT15a] Stefan Borgwardt and Veronika Thost. "Temporal Query Answering in *DL-Lite* with Negation". In: *Proc. of the 1st Global Conference on Artificial Intelligence, GCAI'15*. Ed. by Georg Gottlob, Geoff Sutcliffe, and Andrei Voronkov. Vol. 26. EPiC Series in Computing. EasyChair, 2015, pp. 51–65.

[BT15b] Stefan Borgwardt and Veronika Thost. "Temporal Query Answering in the Description Logic $\mathcal{EL}$". In: *Proc. of the 24th Int. Joint Conference on Artificial Intelligence (IJCAI'15)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 2819–2825.

[BT15c] Stefan Borgwardt and Veronika Thost. "Temporal Query Answering in the Description Logic $\mathcal{EL}$ (extended abstract)". In: *Proc. of the 28th Int. Workshop on Description Logics (DL'15)*. Ed. by Diego Calvanese and Boris Konev. Vol. 1350. CEUR Workshop Proceedings. 2015, pp. 83–87.

[Bus03] Christoph Bussler. “The Role of Semantic Web Technology in Enterprise Application Integration”. In: *IEEE Data Engineering Bulletin* 26.4 (2003), pp. 62–68.

[BV81] Catriel Beeri and Moshe Y. Vardi. “The Implication Problem for Data Dependencies”. In: *Proc. of the 8th Int. Colloquium on Automata, Languages and Programming (ICALP’81)*. Ed. by Shimon Even and Oded Kariv. Vol. 115. Lecture Notes in Computer Science. SV, 1981, pp. 73–85.

[BV84] Catriel Beeri and Moshe Y. Vardi. “A Proof Procedure for Data Dependencies”. In: *Journal of the ACM* 31.4 (1984), pp. 718–741.

[Cal+04] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Riccardo Rosati, and Guido Vetere. “DL-Lite: Practical Reasoning for Rich DLs”. In: *Proc. of the 2004 Int. Workshop on Description Logics (DL’04)*. Ed. by Volker Haarslev and Ralf Möller. Vol. 104. CEUR Workshop Proceedings. 2004, pp. 92–99.

[Cal+05] Diego Calvanese, Giuseppe De Giacomo, Domenico Lemho, Maurizio Lenzerini, and Riccardo Rosati. “*DL-Lite*: Tractable Description Logics for Ontologies”. In: *Proc. of the 20th Nat. Conference on Artificial Intelligence (AAAI’05)*. Ed. by Manuela M. Veloso and Subbaro Kambhampati. AAAI Press, 2005, pp. 602–607.

[Cal+06] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. “Data Complexity of Query Answering in Description Logics”. In: *Proc. of the 10th Int. Conference of Knowledge Representation and Reasoning (KR’06)*. Ed. by Patrick Doherty, John Mylopoulos, and Christopher Welty. AAAI Press, 2006, pp. 260–270.

[Cal+07a] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. “EQL-Lite: Effective First-Order Query Processing in Description Logics”. In: *Proc. of the 20th Int. Joint Conference on Artificial Intelligence (IJCAI’07)*. Ed. by Manuela M. Veloso. 2007, pp. 274–279.

[Cal+07b] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. “Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family”. In: *Journal of Automated Reasoning* 39.3 (2007), pp. 385–429.

[Cal+17] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. “Ontop: Answering SPARQL Queries over Relational Databases”. In: *Semantic Web* 8.3 (2017), pp. 471–487.

[CCG10] Jean-Paul Calbimonte, Oscar Corcho, and Alasdair J. G. Gray. “Enabling Ontology-based Access to Streaming Data Sources”. In: *Proc. of the 9th Int. Semantic Web Conference (ISWC’10), Part I*. Ed. by Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian

Horrocks, and Birte Glimm. Vol. 6496. Lecture Notes in Computer Science. Springer-Verlag, 2010, pp. 96–111.

[CE81] Edmund M. Clarke and E. Allen Emerson. "Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic". In: *Logics of Programs, Workshop*. Ed. by Dexter Kozen. Vol. 131. Lecture Notes in Computer Science. Springer-Verlag, 1981, pp. 52–71.

[CGL09] Andrea Calí, Georg Gottlob, and Thomas Lukasiewicz. "A General Datalog-Based Framework for Tractable Query Answering over Ontologies". In: *Proc. of the 28th Symposium on Principles of Database Systems (PODS'09)*. Ed. by Jan Paredaens and Jianwen Su. ACM, 2009, pp. 77–86.

[CGL12] Andrea Calí, Georg Gottlob, and Thomas Lukasiewicz. "A General Datalog-Based Framework for Tractable Query Answering over Ontologies". In: *Journal of Web Semantics* 14 (2012), pp. 57–83.

[CGP12] Andrea Calí, Georg Gottlob, and Andreas Pieris. "Towards More Expressive Ontology Languages: The Query Answering Problem". In: *Journal of Artificial Intelligence* 193 (2012), pp. 87–128.

[Che76] Peter Pin-Shan Chen. "The Entity-Relationship Model—Toward a Unified View of Data". In: *ACM Transactions on Database Systems* 1.1 (1976), pp. 9–36. ISSN: 0362-5915.

[Cho95] Jan Chomicki. "Efficient Checking of Temporal Integrity Constraints Using Bounded History Encoding". In: *ACM Transactions on Database Systems* 20.2 (1995), pp. 148–186.

[CKS81] Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. "Alternation". In: *Journal of the ACM* 28.1 (1981), pp. 114–133.

[CLM81] Ashok K. Chandra, Harry R. Lewis, and Johann A. Makowsky. "Embedded Implicational Dependencies and their Inference Problem". In: *Proc. of the 13th Annual Symposium on Theory of Computation (STOC'81)*. ACM, 1981, pp. 342–354.

[CM77] Ashok K. Chandra and Philip M. Merlin. "Optimal Implementation of Conjunctive Queries in Relational Data Bases". In: *Proc. of the 9th Annual Symposium on Theory of Computing (STOC'77)*. ACM, 1977, pp. 77–90.

[CMC16] Jean-Paul Calbimonte, José Mora, and Óscar Corcho. "Query Rewriting in RDF Stream Processing". In: *Proc. of the 13th Int. Semantic Web Conference (ESWC'16)*. Ed. by Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange. Vol. 9678. Lecture Notes in Computer Science. Springer-Verlag, 2016, pp. 486–502.

[Com+12] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcıa-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. "The SSN Ontology of the W3C Semantic Sensor Network Incubator

Group". In: *Journal of Web Semantics* 17 (2012), pp. 25–32. ISSN: 1570-8268.

[CR15] Cristina Civili and Riccardo Rosati. "On the First-Order Rewritability of Conjunctive Queries over Binary Guarded Existential Rules". In: *Proc. of the 30th Italian Conference on Computational Logic*. Ed. by Davide Ancona, Marco Maratea, and Viviana Mascardi. Vol. 1459. CEUR Workshop Proceedings. 2015, pp. 25–30.

[CT05] Jan Chomicki and David Toman. "Temporal Databases". In: *Handbook of Temporal Reasoning in Artificial Intelligence*. Ed. by Michael Fisher, Dov M. Gabbay, and Lluis Vila. Elsevier Science Inc., 2005, pp. 429–467.

[CTB01] Jan Chomicki, David Toman, and Michael H. Böhlen. "Querying ATSQL Databases with Temporal Logic". In: *ACM Transactions on Database Systems* 26.2 (2001), pp. 145–178.

[CWL14] Richard Cyganiak, David Wood, and Markus Lanthaler, eds. *RDF 1.1 Concepts and Abstract Syntax*. Available at `http://www.w3.org/TR/rdf11-concepts/`. W3C Recommendation, 25 February 2014.

[Dar+13] Waltenegus Dargie, Eldora, Julian Mendez, Christoph Möbius, Kateryna Rybina, Veronika Thost, and Anni-Yasmin Turhan. "Situation Recognition for Service Management Systems Using OWL 2 Reasoners". In: *Proc. of the 10th Workshop on Context Modeling and Reasoning 2013*. IEEE Computer Society, 2013, pp. 31–36.

[DDM16] Emanuele Della Valle, Daniele Dell'Aglio, and Alessandro Margara. "Taming Velocity and Variety Simultaneously in Big Data with Stream Reasoning: Tutorial". In: *Proc. of the 10th Int. Conference on Distributed and Event-based Systems (DEBS'16)*. Ed. by Avigdor Gal, Matthias Weidlich, Vana Kalogeraki, and Nalini Venkasubramanian. ACM, 2016, pp. 394–401.

[Del+15] Daniele Dell'Aglio, Jean-Paul Calbimonte, Emanuele Della Valle, and Óscar Corcho. "Towards a Unified Language for RDF Stream Query Processing". In: *The Semantic Web: ESWC 2015 Satellite Events, Revised Selected Papers*. Ed. by Fabien Gandon, Christophe Guéret, Serena Villata, John Breslin, Catherine Faron-Zucker, and Antoine Zimmermann. Vol. 9341. Lecture Notes in Computer Science. Springer-Verlag, 2015, pp. 353–363.

[Del+16] Daniele Dell'Aglio, Minh Dao-Tran, Jean-Paul Calbimonte, Danh Le Phuoc, and Emanuele Della Valle. "A Query Model to Capture Event Pattern Matching in RDF Stream Processing Query Languages". In: *Proc. of the 20th Int. Conference of Knowledge Engineering and Knowledge Management (EKAW 2016)*. Ed. by Eva Blomqvist, Paolo Ciancarini, Francesco Poggi, and Fabio Vitali. Vol. 10024. Lecture Notes in Computer Science. Springer-Verlag, 2016, pp. 145–162.

[Dim+16] Mirko Michele Dimartino, Andrea Calí, Alexandra Poulovassilis, and Peter T. Wood. "Query Rewriting under Linear $\mathcal{EL}$ Knowledge Bases". In: *Proc. of the 10th Int. Conference on Web Reasoning and Rule Systems (RR'16)*. Ed. by Magdalena Ortiz and Stefan Schlobach. Vol. 9898. Lecture Notes in Computer Science. Springer-Verlag, 2016, pp. 61–76.

[DNR08] Alin Deutsch, Alan Nash, and Jeffrey B. Remmel. "The Chase Revisited". In: *Proc. of the 27th Symposium on Principles of Database Systems (PODS'08)*. Ed. by Maurizio Lenzerini and Domenico Lembo. ACM, 2008, pp. 149–158.

[DS04] Mike Dean and Guus Schreiber, eds. *OWL Web Ontology Language Reference*. Available at `http://www.w3.org/TR/owl-ref/`. W3C Recommendation, 10 February 2004.

[Eit+12] Thomas Eiter, Magdalena Ortiz, Mantas Šimkus, Trung-Kien Tran, and Guohui Xiao. "Query Rewriting for Horn-$\mathcal{SHIQ}$ Plus Rules". In: *Proc. of the 26th AAAI Conference on Artificial Intelligence (AAAI'12)*. Ed. by Jörg Hoffmann and Bart Selman. AAAI Press, 2012, pp. 726–733.

[Fit96] Melvin Fitting. *First-order Logic and Automated Theorem Proving*. 2nd ed. Springer-Verlag, 1996.

[FSS84] Merrick L. Furst, James B. Saxe, and Michael Sipser. "Parity, Circuits, and the Polynomial-Time Hierarchy". In: *Mathematical Systems Theory* 17.1 (1984), pp. 13–27.

[Gab87] Dov M. Gabbay. "The Declarative Past and Imperative Future: Executable Temporal Logic for Interactive Systems". In: *Proc. of Temporal Logic in Specification*. Ed. by Behnam Banieqbal, Howard Barringer, and Amir Pnueli. Vol. 398. Lecture Notes in Computer Science. Springer-Verlag, 1987, pp. 409–448.

[GHV07] Claudio Gutierrez, Carlos A. Hurtado, and Alejandro A. Vaisman. "Introducing Time into RDF". In: *IEEE Trans. Knowl. Data Eng.* 19.2 (2007), pp. 207–218.

[GJK16] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Roman Kontchakov. "Temporalized $\mathcal{EL}$ Ontologies for Accessing Temporal Data: Complexity of Atomic Queries". In: *Proc. of the 25th Int. Joint Conference on Artificial Intelligence (IJCAI'16)*. Ed. by Subbarao Kambhampati. AAAI Press, 2016, pp. 1102–1108.

[GJL12] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Carsten Lutz. "Complexity of Branching Temporal Description Logics". In: *Proc. of the 20th European Conference on Artificial Intelligence (ECAI'12)*. Ed. by Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas. Vol. 242. Frontiers in Artificial Intelligence and Applications. IOS Press, 2012, pp. 390–395.

[GJO16] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Ana Ozaki. "On Metric Temporal Description Logics". In: *Proc. of the 22nd European Conference on Artificial Intelligence (ECAI'16)*. Ed. by Gal A. Kaminka, Maria Fox, Paolo Bouquet, Eyke Hüllermeier, Virginia Dignum, Frank Dignum, and Frank van Harmelen. Vol. 285. Frontiers in Artificial Intelligence and Applications. IOS Press, 2016, pp. 837–845.

[GJS14] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Thomas Schneider. "Lightweight Description Logics and Branching Time: A Troublesome Marriage". In: *Proc. of the 14th Int. Conference of Knowledge Representation and Reasoning (KR'14)*. Ed. by Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter. AAAI Press, 2014, pp. 278–287.

[GJS15] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Thomas Schneider. "Lightweight Temporal Description Logics with Rigid Roles and Restricted TBoxes". In: *Proc. of the 24th Int. Joint Conference on Artificial Intelligence (IJCAI'15)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 3015–3021.

[GK12] Víctor Gutiérrez-Basulto and Szymon Klarman. "Towards a Unifying Approach to Representing and Querying Temporal Data in Description Logics". In: *Proc. of the 6th Int. Conference on Web Reasoning and Rule Systems (RR'12)*. Ed. by Markus Krötzsch and Umberto Straccia. Vol. 7497. Lecture Notes in Computer Science. Springer-Verlag, 2012, pp. 90–105.

[Gli+08] Birte Glimm, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. "Conjunctive Query Answering for the Description Logic $\mathcal{SHIQ}$". In: *Journal of Artificial Intelligence Research* 31.1 (2008), pp. 157–204.

[GOP11] Georg Gottlob, Giorgio Orsi, and Andreas Pieris. "Ontological Queries: Rewriting and Optimization". In: *Proc. of the 27th Int. Conference on Data Engineering (ICDE'11)*. Ed. by Serge Abiteboul, Klemens Böhm, Christoph Koch, and Kian-Lee Tan. IEEE Press, 2011, pp. 2–13.

[Gut+15] Víctor Gutiérrez-Basulto, Yazmin Angélica Ibáñez-García, Roman Kontchakov, and Egor V. Kostylev. "Queries with Negation and Inequalities over Lightweight Ontologies". In: *Journal of Web Semantics* 35 (2015), pp. 184–202.

[Häh+14] Marcus Hähnel, Julian Mendez, Veronika Thost, and Anni-Yasmin Turhan. "Bridging the Application Knowledge Gap". In: *Proc. of the 13th Workshop on Adaptive and Reflective Middleware (ARM@Middleware'14)*. Ed. by Fábio M. Costa and Anders Andersen. ACM, 2014, 3:1–3:6.

[Han+15] Peter Hansen, Carsten Lutz, Inanç Seylan, and Frank Wolter. "Efficient Query Rewriting in the Description Logic $\mathcal{EL}$ and Beyond". In: *Proc. of the 24th Int. Joint Conference on Artificial Intelligence (IJCAI'15)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 3034–3040.

[Har+13] Babak Bagheri Hariri, Diego Calvanese, Marco Montali, Giuseppe De Giacomo, Riccardo De Masellis, and Paolo Felli. "Description Logic Knowledge and Action Bases". In: *Journal of Artificial Intelligence Research* 46 (2013), pp. 651–686.

[HEX10] Stijn Heymans, Thomas Eiter, and Guohui Xiao. "Tractable Reasoning with DL-Programs over Datalog-rewritable Description Logics". In: *Proc. of the 19th European Conference on Artificial Intelligence (ECAI'10)*. Ed. by Helder Coelho, Rudi Studer, and Michael Wooldridge. Vol. 215. Frontiers in Artificial Intelligence and Applications. IOS Press, 2010, pp. 35–40.

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. "The Even More Irresistible $\mathcal{SROIQ}$". In: *Proc. of the 10th Int. Conference of Knowledge Representation and Reasoning (KR'06)*. Ed. by Patrick Doherty, John Mylopoulos, and Christopher Welty. AAAI Press, 2006, pp. 57–67.

[HMS07] Ullrich Hustadt, Boris Motik, and Ulrike Sattler. "Reasoning in Description Logics by a Reduction to Disjunctive Datalog". In: *Journal of Automated Reasoning* 39.3 (2007), pp. 351–384.

[HS91a] Joseph Y. Halpern and Yoav Shoham. "A Propositional Modal Logic of Time Intervals". In: *Journal of the ACM* 38.4 (1991), pp. 935–962.

[HS91b] Klaus Hülsmann and Gunter Saake. "Theoretical Foundations of Handling Large Substitution Sets in Temporal Integrity Monitoring". In: *Acta Informatica* 28.4 (1991), pp. 365–407.

[Kar72] Richard Karp. "Reducibility among Combinatorial Problems". In: *Proc. of a Symposium on the Complexity of Computer Computations*. Ed. by Raymond E. Miller and James W. Thatcher. Plenum Press, 1972, pp. 85–103.

[Kaz09] Yevgeny Kazakov. "Consequence-Driven Reasoning for Horn $\mathcal{SHIQ}$ Ontologies". In: *Proc. of the 21st Int. Joint Conference on Artificial Intelligence (IJCAI'09)*. Ed. by Craig Boutilier. AAAI Press, 2009, pp. 2040–2045.

[Kha+15] Evgeny Kharlamov, Dag Hovland, Ernesto Jiménez-Ruiz, Davide Lanti, Hallstein Lie, Christoph Pinkel, Martin Rezk, Martin G. Skjæveland, Evgenij Thorstensen, Guohui Xiao, Dmitriy Zheleznyakov, and Ian Horrocks. "Ontology Based Access to Exploration Data at Statoil". In: *Proc. of the 14th Int. Semantic Web Conference (ISWC'15), Part II*. Ed. by Marcelo Arenas, Óscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul T. Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab. Vol. 9366. Lecture Notes in Computer Science. Springer-Verlag, 2015, pp. 93–112.

[Kha+16] Evgeny Kharlamov, Yannis Kotidis, Theofilos Mailis, Christian Neuenstadt, Charalampos Nikolaou, Özgür L. Özçep, Christoforos Svingos, Dmitriy Zheleznyakov, Sebastian Brandt, Ian Horrocks, Yannis E. Ioannidis, Steffen Lamparter, and Ralf Möller. "Towards Analytics Aware Ontology

Based Access to Static and Streaming Data". In: *Proc. of the 15th Int. Semantic Web Conference (ISWC'16), Part II*. Ed. by Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil. Vol. 9981. Lecture Notes in Computer Science. Springer-Verlag, 2016, pp. 344–362.

[KKS12] Yevgeny Kazakov, Markus Krötzsch, and František Simančík. "Practical Reasoning with Nominals in the $\mathcal{EL}$ Family of Description Logics". In: *Proc. of the 13th Int. Conference of Knowledge Representation and Reasoning (KR'12)*. Ed. by Gerhard Brewka, Thomas Eiter, and Sheila A. McIlraith. AAAI Press, 2012, pp. 264–274.

[KL07] Adila Krisnadhi and Carsten Lutz. "Data Complexity in the $\mathcal{EL}$ Family of Description Logics". In: *Proc. of the 14th Int. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'07)*. Ed. by Nachum Dershowitz and Andrei Voronkov. Vol. 4790. Lecture Notes in Computer Science. Springer-Verlag, 2007, pp. 333–347.

[Kla13] Szymon Klarman. "Practical Querying of Temporal Data via OWL 2 QL and SQL: 2011". In: *LPAR 2013, 19th Int. Conference on Logic for Programming, Artificial Intelligence and Reasoning, Short papers proceedings*. Ed. by Kenneth L. McMillan, Aart Middeldorp, Geoff Sutcliffe, and Andrei Voronkov. Vol. 26. EPiC Series in Computing. EasyChair, 2013, pp. 52–61.

[KM12] Krishna G. Kulkarni and Jan-Eike Michels. "Temporal features in SQL: 2011". In: *SIGMOD Record* 41.3 (2012), pp. 34–43.

[KM13] Szymon Klarman and Thomas Meyer. "Prediction and Explanation over DL-Lite Data Streams". In: *Proc. of the 19th Int. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'13)*. Ed. by Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov. Vol. 8312. Lecture Notes in Computer Science. Springer-Verlag, 2013, pp. 536–551.

[KM14a] Szymon Klarman and Thomas Meyer. "Complexity of Temporal Query Abduction in *DL-Lite*". In: *Proc. of the 27th Int. Workshop on Description Logics (DL'14)*. Ed. by Meghyn Bienvenu, Magdalena Ortiz, Riccardo Rosati, and Mantas Šimkus. Vol. 1193. CEUR Workshop Proceedings. 2014, pp. 233–244.

[KM14b] Szymon Klarman and Thomas Meyer. "Querying Temporal Databases via OWL 2 QL". In: *Proc. of the 8th Int. Conference on Web Reasoning and Rule Systems (RR'14)*. Ed. by Roman Kontchakov and Marie-Laure Mugnier. Vol. 8741. Lecture Notes in Computer Science. Springer-Verlag, 2014, pp. 92–107.

[Kon+10] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. "The Combined Approach to Query Answering in *DL-Lite*". In: *Proc. of the 12th Int. Conference of Knowledge Representation and Reasoning (KR'10)*. Ed. by Fangzhen Lin, Ulrike Sattler, and Miroslaw Truszczynski. AAAI Press, 2010, pp. 247–257.

[Kon+11] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. "The Combined Approach to Ontology-Based Data Access". In: *Proc. of the 22nd Int. Joint Conference on Artificial Intelligence (IJCAI'11)*. Ed. by Toby Walsh. AAAI Press, 2011, pp. 2656–2661.

[Kon+16] Roman Kontchakov, Laura Pandolfo, Luca Pulina, Vladislav Ryzhikov, and Michael Zakharyaschev. "Temporal and Spatial OBDA with Many-Dimensional Halpern-Shoham Logic". In: *Proc. of the 25th Int. Joint Conference on Artificial Intelligence (IJCAI'16)*. Ed. by Subbarao Kambhampati. AAAI Press, 2016, pp. 1160–1166.

[KRH07] Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler. "Conjunctive Queries for a Tractable Fragment of OWL 1.1". In: *Proc. of the 6th Int. Semantic Web Conference (ISWC'07)*. Ed. by Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Goldbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux. Vol. 4825. Lecture Notes in Computer Science. Springer-Verlag, 2007, pp. 310–323.

[KRH13] Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler. "Complexities of Horn Description Logics". In: *ACM Transactions on Computational Logic* 14.1 (2013), 2:1–2:36.

[Krö11] Markus Krötzsch. "Efficient Rule-Based Inferencing for OWL EL". In: *Proc. of the 22nd Int. Joint Conference on Artificial Intelligence (IJCAI'11)*. Ed. by Toby Walsh. AAAI Press, 2011, pp. 2668–2773.

[Kur+03] Agi Kurucz, Frank Wolter, Michael Zakharyaschev, and Dov M. Gabbay. *Many-Dimensional Modal Logics: Theory and Applications*. Vol. 148. Gulf Professional Publishing, 2003.

[Len02] Maurizio Lenzerini. "Data Integration: A Theoretical Perspective". In: *Proc. of the 21st Symposium on Principles of Database Systems (PODS'02)*. Ed. by Lucian Popa, Serge Abiteboul, and Phokion G. Kolaitis. ACM, 2002, pp. 233–246.

[Li+13] Jianwen Li, Lijun Zhang, Geguang Pu, Moshe Y. Vardi, and Jifeng He. "LTL Satisfiability Checking Revisited". In: *Proc. of the 20th Int. Symposium on Temporal Representation and Reasoning (TIME'13)*. Ed. by César Sánchez, Kristen Brent Venable, and Esteban Zimányi. IEEE Computer Society, 2013, pp. 91–98.

[Li+15] Jianwen Li, Shufang Zhu, Geguang Pu, and Moshe Y. Vardi. "SAT-Based Explicit LTL Reasoning". In: *Proc. of the 11th Int. Haifa Verification Conference (HVC'15)*. Ed. by Nir Piterman. Vol. 9434. Lecture Notes in Computer Science. Springer-Verlag, 2015, pp. 209–224.

[Lip14] Marcel Lippmann. "Temporalised Description Logics for Monitoring Partially Observable Events". 2014. URL: `http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-147977`.

[LTW09] Carsten Lutz, David Toman, and Frank Wolter. "Conjunctive Query Answering in the Description Logic $\mathcal{EL}$ using a Relational Database System". In: *Proc. of the 21st Int. Joint Conference on Artificial Intelligence (IJCAI'09)*. Ed. by Craig Boutilier. AAAI Press, 2009, pp. 2070–2075.

[Lut01] Carsten Lutz. "Interval-Based Temporal Reasoning with General TBoxes". In: *Proc. of the 17th Int. Joint Conference on Artificial Intelligence (IJCAI'01)*. Ed. by Bernhard Nebel. Morgan Kaufmann, 2001, pp. 89–96.

[LWW07] Carsten Lutz, Dirk Walther, and Frank Wolter. "Quantitative Temporal Logics over the Reals: PSpace and Below". In: *Inf. Comput.* 205.1 (2007), pp. 99–123. ISSN: 0890-5401.

[LWZ08] Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. "Temporal Description Logics: A Survey". In: *Proc. of the 15th Int. Symposium on Temporal Representation and Reasoning (TIME'08)*. Ed. by Stéphane Demri and Christian S. Jensen. IEEE Press, 2008, pp. 3–14.

[Min74] Marvin Minsky. *A Framework for Representing Knowledge*. Artificial Intelligence Memo. Cambridge, MA, USA: A.I. Laboratory, Massachusetts Institute of Technology, 1974.

[Mot+12] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz, eds. *OWL 2 Web Ontology Language: Profiles (Second Edition)*. Available at `http://www.w3.org/TR/owl2-profiles/`. W3C Recommendation 11 December 2012, 2012.

[Mot12] Boris Motik. "Representing and Querying Validity Time in RDF and OWL: A Logic-Based Approach". In: *Journal of Web Semantics* 12–13 (2012), pp. 3–21.

[OMG15] Object Management Group (OMG), ed. *OMG Unified Modeling Language (OMG UML) Version 2.5*. Available at `http://www.omg.org/spec/UML/2.5/PDF`. March 2015.

[Ont15] Ontologies for Robotics and Automation Working Group. "IEEE Standard Ontologies for Robotics and Automation". In: *IEEE Std 1872-2015* (2015), pp. 1–60.

[ORŠ11] Magdalena Ortiz, Sebastian Rudolph, and Mantas Šimkus. "Query Answering in the Horn Fragments of the Description Logics $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$". In: *Proc. of the 22nd Int. Joint Conference on Artificial Intelligence (IJCAI'11)*. Ed. by Toby Walsh. AAAI Press, 2011, pp. 1039–1044.

[OWL12] W3C OWL Working Group. *OWL 2 Web Ontology Language: Document Overview (Second Edition)*. Available at `http://www.w3.org/TR/owl2-overview/`. W3C Recommendation 11 December 2012, 2012.

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.

[PK09] Adrian Paschke and Alexander Kozlenkov. “Rule-Based Event Processing and Reaction Rules”. In: *Proc. of the 2009 Int. Symposium on Rule Interchange and Applications (RuleML'09)*. Ed. by Guido Governatori, John Hall, and Adrian Paschke. Vol. 5858. Lecture Notes in Computer Science. Springer-Verlag, 2009, pp. 53–66.

[PMH10] Héctor Pérez-Urbina, Boris Motik, and Ian Horrocks. “Tractable Query Answering and Rewriting under Description Logic Constraints”. In: *Journal of Applied Logic* 8.2 (2010), pp. 186–209.

[Pnu77] Amir Pnueli. “The Temporal Logic of Programs”. In: *Proc. of the 18th Annual Symposium on Foundations of Computer Science (FOCS'77)*. IEEE Press, 1977, pp. 46–57.

[Qui67] M. Ross Quillian. “Word Concepts: A Theory and Simulation of Some Basic Semantic Capabilities.” In: *Behavioral Science* 12.5 (1967), pp. 410–430. ISSN: 0005-7940.

[RA10] Riccardo Rosati and Alessandro Almatelli. “Improving Query Answering over *DL-Lite* Ontologies”. In: *Proc. of the 12th Int. Conference of Knowledge Representation and Reasoning (KR'10)*. Ed. by Fangzhen Lin, Ulrike Sattler, and Miroslaw Truszczynski. AAAI Press, 2010, pp. 290–300.

[Rec+94] Alan Rector, Also Gangemi, Elena Galeazzi, Andrzej J. Glowinski, and Aangelo Rossi-Mori. “The GALEN CORE Model Schemata for Anatomy: Towards a Re-usable Application-Independent Model of Medical Concepts”. In: *Proc. of Medical Informatics Europe (MIE94)*. 1994, pp. 186–189.

[RG10] Sebastian Rudolph and Birte Glimm. “Nominals, Inverses, Counting, and Conjunctive Queries or: Why Infinity is your Friend!” In: *Journal of Artificial Intelligence Research* 39.1 (2010), pp. 429–481.

[Ros07] Riccardo Rosati. “On Conjunctive Query Answering in $\mathcal{EL}$”. In: *Proc. of the 2007 Int. Workshop on Description Logics (DL'07)*. Ed. by Diego Calvanese, Enrico Franconi, Volker Haarslev, Domenico Lembo, Boris Motik, Anni-Yasmin Turhan, and Sergio Tessaris. Vol. 250. CEUR Workshop Proceedings. 2007, pp. 451–458.

[Sav70] Walter J. Savitch. “Relationships Between Nondeterministic and Deterministic Tape Complexities”. In: *Journal of Computer and System Sciences* 4.2 (1970), pp. 177–192.

[SC85] A. Prasad Sistla and Edmund M. Clarke. “The Complexity of Propositional Linear Temporal Logics”. In: *Journal of the ACM* 32.3 (1985), pp. 733–749.

[SCC97] Kent A. Spackman, Keith E. Campbell, and Roger A. Côté. “SNOMED RT: A Reference Terminology for Health Care”. In: *Proc. of the AMIA Annual Fall Symposium*. AMIA, 1997, pp. 640–644.

[Sch93] Klaus Schild. “Combining Terminological Logics with Tense Logic”. In: *Proc. of the 6th Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence (EPIA '93)*. Springer-Verlag, 1993, pp. 105–120.

[SL89] Gunter Saake and Udo W. Lipeck. “Using Finite-Linear Temporal Logic for Specifying Database Dynamics”. In: *Proc. of the 2nd Workshop on Computer Science Logic (CSL'88)*. Ed. by Egon Börger, Hans Kleine Büning, and Michael M. Richter. Vol. 385. Lecture Notes in Computer Science. Springer-Verlag, 1989, pp. 288–300.

[SS91] Manfred Schmidt-Schauß and Gert Smolka. “Attributive Concept Descriptions with Complements”. In: *Journal of Artificial Intelligence* 48.1 (1991), pp. 1–26.

[The13] The W3C SPARQL Working Group, ed. *SPARQL 1.1 Overview*. Available at `http://www.w3.org/TR/sparql11-overview/`. W3C Recommendation 21 March 2013, 2013.

[THÖ15] Veronika Thost, Jan Holste, and Özgür L. Özçep. “On Implementing Temporal Query Answering in *DL-Lite* (extended abstract)”. In: *Proc. of the 28th Int. Workshop on Description Logics (DL'15)*. Ed. by Diego Calvanese and Boris Konev. Vol. 1350. CEUR Workshop Proceedings. 2015, pp. 552–555.

[Tom04] David Toman. “Logical Data Expiration”. In: *Logics for Emerging Applications of Databases*. Ed. by Jan Chomicki, Ron van der Meyden, and Gunter Saake. Springer-Verlag, 2004, pp. 203–238.

[VW86] Moshe Y. Vardi and Pierre Wolper. “Automata-Theoretic Techniques for Modal Logics of Programs”. In: *Journal of Computer and System Sciences* 32.2 (1986), pp. 183–221.

[Wil99] Thomas Wilke. “Classifying Discrete Temporal Properties”. In: *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS'99)*. Ed. by Christoph Meinel and Sophie Tison. Vol. 1563. Lecture Notes in Computer Science. Springer-Verlag, 1999, pp. 32–46.

[Wul+08] Martin De Wulf, Laurent Doyen, Nicolas Maquet, and Jean-François Raskin. “Antichains: Alternative Algorithms for LTL Satisfiability and Model-Checking”. In: *Proc. of the 14th Int. Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'08)*. Ed. by C. R. Ramakrishnan and Jakob Rehof. Vol. 4963. Lecture Notes in Computer Science. Springer-Verlag, 2008, pp. 63–77.

Index