Journal of Artificial Intelligence Research 51 (2014) 645-705 Submitted 05/14; published 12/14

The Complexity of Answering Conjunctive and Navigational Queries over OWL 2 EL Knowledge Bases

Giorgio Stefanoni GIORGIO.STEFANONI@CS.OX.AC.UK
Boris Motik BORIS.MOTIK@CS.OX.AC.UK
Department of Computer Science, University of Oxford
Parks Road, Oxford OX1 3QD, United Kingdom

Markus Krötzsch MARKUS.KROETZSCH@TU-DRESDEN.DE
Sebastian Rudolph SEBASTIAN.RUDOLPH@TU-DRESDEN.DE
Faculty of Computer Science, TU Dresden
Nöthnitzer Straße 46, 01062 Dresden, Germany

Abstract

OWL 2 EL is a popular ontology language that supports *role inclusions*—axioms of the form $S_1 \cdots S_n \sqsubseteq S$ that capture compositional properties of roles. Role inclusions closely correspond to context-free grammars, which was used to show that answering conjunctive queries (CQs) over OWL 2 EL knowledge bases with unrestricted role inclusions is undecidable. However, OWL 2 EL inherits from OWL 2 DL the syntactic *regularity* restriction on role inclusions, which ensures that role chains implying a particular role can be described using a finite automaton (FA). This is sufficient to ensure decidability of CQ answering; however, the FAs can be worst-case exponential in size so the known approaches do not provide a tight upper complexity bound.

In this paper, we solve this open problem and show that answering CQs over OWL 2 EL knowledge bases is PSPACE-complete in combined complexity (i.e., the complexity measured in the total size of the input). To this end, we use a novel encoding of regular role inclusions using *bounded-stack pushdown automata*—that is, FAs extended with a stack of bounded size. Apart from theoretical interest, our encoding can be used in practical tableau algorithms to avoid the exponential blowup due to role inclusions. In addition, we sharpen the lower complexity bound and show that the problem is PSPACE-hard even if we consider only role inclusions as part of the input (i.e., the query and all other parts of the knowledge base are fixed). Finally, we turn our attention to navigational queries over OWL 2 EL knowledge bases, and we show that answering positive, converse-free conjunctive graph XPath queries is PSPACE-complete as well; this is interesting since allowing the converse operator in queries is known to make the problem EXPTIME-hard. Thus, in this paper we present several important contributions to the landscape of the complexity of answering expressive queries over description logic knowledge bases.

1. Introduction

Description logics (DLs) (Baader, Calvanese, McGuinness, Nardi, & Patel-Schneider, 2010) are a family of knowledge representation formalisms that logically underpin the *Web Ontology Language* OWL 2 (Cuenca Grau, Horrocks, Motik, Parsia, Patel-Schneider, & Sattler, 2008). DL knowledge bases describe a domain in terms of *concepts* (i.e., unary predicates), *roles* (i.e., binary predicates), and *individuals* (i.e., constants), and they describe the relationships between concepts, roles, and individuals using logical *axioms*. DLs and OWL

2 have been steadily gaining in popularity because they provide the developers of modern information systems with a flexible graph-like data model that is natural in countless application areas, such as the Semantic Web (Gutierrez, Hurtado, Mendelzon, & Pérez, 2011), social network analysis (Fan, 2012), and network traffic analysis (Barrett, Jacob, & Marathe, 2000). Answering queries over DL/OWL knowledge bases is the core service in applications as diverse as monitoring financial products within the Italian Ministry of Economy and Finance (De Giacomo et al., 2012), accessing real-time diagnostic data of turbines (Giese et al., 2013), and integrating configuration data of air traffic control systems (Calvanese et al., 2011). Due to the practical importance of query answering, theoretical investigation of the expressivity and computational complexity of query languages has been high up on the research agenda of the knowledge representation community in the past decade.

Conjunctive queries (CQs) (Chandra & Merlin, 1977) are the basic class of queries in relational databases. Querying DL knowledge bases using CQs has been studied in a diverse range of settings (Calvanese, De Giacomo, Lembo, Lenzerini, & Rosati, 2007; Pérez-Urbina, Motik, & Horrocks, 2010; Rudolph & Glimm, 2010; Kontchakov, Lutz, Toman, Wolter, & Zakharyaschev, 2011; Ortiz, Rudolph, & Simkus, 2011; Gottlob & Schwentick, 2012; Venetis, Stoilos, & Stamou, 2012). However, conjunctive queries are first-order definable and thus cannot express certain important properties such as graph reachability. *Regular path queries* (RPQs) (Cruz, Mendelzon, & Wood, 1987; Barceló, 2013) are an alternative query language capable of describing connections between graph vertices using regular expressions, allowing users to 'navigate' inside a graph. For example, the RPQ $(\mathsf{isPartOf}^* \cdot \mathsf{hasLocation})$ retrieves all pairs of vertices connected via zero or more $\mathsf{isPartOf}$ edges followed by one $\mathsf{hasLocation}$ edge. Furthermore, 2RPQs extend RPQs with the *converse operator* (i.e., backward navigation) (Calvanese, Vardi, De Giacomo, & Lenzerini, 2000); nested regular expressions allow for existential quantification over paths (Pérez, Arenas, & Gutierrez, 2010); and C(2)RPQs extend both (2)RPQs and CQs to conjunctions of (2)RPQs (Calvanese, De Giacomo, Lenzerini, & Vardi, 2000; Bienvenu, Ortiz, & Simkus, 2013). Finally, inspired by the XPath query language for XML, *graph XPath queries* (GXQs) have been recently proposed as a language for querying graph databases (Libkin, Martens, & Vrgoč, 2013) and DL knowledge bases (Kostylev, Reutter, & Vrgoc, 2014; Bienvenu, Calvanese, Ortiz, & Simkus, 2014). GXQs extend 2RPQs with negation on regular expressions, and checking properties of vertices using Boolean combinations of *node tests*—that is, concepts or existential quantifications over paths. For example, the graph XPath query $(\mathsf{isPartOf}^* \cdot \mathsf{test}(\mathsf{Cell} \wedge \neg\langle\mathsf{hasSpeciality}\rangle) \cdot \mathsf{hasLocation})$ refines the aforementioned RPQ by requiring that the node between the $\mathsf{isPartOf}$ edges and the $\mathsf{hasLocation}$ edge is an instance of the Cell concept and does *not* have an outgoing $\mathsf{hasSpeciality}$ edge. Graph XPath queries can be straightforwardly extended to *conjunctive graph XPath queries* (CGXQs). A query in any of these languages is *Boolean* if it has no answer variables; hence, an answer to such a query is a Boolean value.

1.1 Problem Setting

Although computing answers to a query over a DL knowledge base is a function problem, it is common in the literature to consider the complexity of the associated decision problem—that is, of checking whether a Boolean query is entailed by the knowledge base. In this article we follow this well-established practice and analyse the computational properties of several query languages over DL knowledge bases. We follow Vardi (1982) and measure the input size in two ways: *combined complexity* measures the complexity in terms of the combined size of the query and the knowledge base, while *data complexity* measures the complexity in terms of the size of the data (i.e., the query and all other parts of the knowledge bases are considered to be fixed).

The computational properties of query answering over DL knowledge bases depend on the expressivity of both the constructs used in the knowledge base and the query language used. In particular, conjunctive query answering over expressive description logics is at least exponential in combined complexity (Glimm, Lutz, Horrocks, & Sattler, 2008; Lutz, 2008) and intractable in data complexity (Calvanese, De Giacomo, Lembo, Lenzerini, & Rosati, 2013; Ortiz, Calvanese, & Eiter, 2008). The problem becomes tractable in data complexity for the RL (Grosof, Horrocks, Volz, & Decker, 2003; ter Horst, 2005) and the QL (Calvanese et al., 2007; Artale, Calvanese, Kontchakov, & Zakharyaschev, 2009) profiles of OWL 2, and several worst-case optimal algorithms have been proposed that perform well in practice (Urbani, van Harmelen, Schlobach, & Bal, 2011; Rodriguez-Muro & Calvanese, 2012). In this paper, however, we focus on the OWL 2 EL profile of OWL 2, which is based on the $\mathcal{EL}$ family of DLs (Baader, Brandt, & Lutz, 2005). Basic reasoning problems for OWL 2 EL, such as checking concept subsumption and instance checking, can be decided in polynomial time (Baader et al., 2005; Krötzsch, 2011), which makes this language very interesting for practical applications. Motivated by this observation, in this paper we present several novel complexity results for answering queries over OWL 2 EL knowledge bases.

One of the important modelling constructs of OWL 2 EL are *role inclusions*—axioms of the form $S_1 \cdots S_n \sqsubseteq S$ that express compositional properties of roles. For example, the following inclusions state that role $\mathsf{isPartOf}$ is transitive and that, if x is located in y and y is part of z, then x is located in z.

$$\mathsf{isPartOf} \cdot \mathsf{isPartOf} \sqsubseteq \mathsf{isPartOf} \qquad \mathsf{hasLocation} \cdot \mathsf{isPartOf} \sqsubseteq \mathsf{hasLocation}$$

Prior to the introduction of the $\mathcal{EL}$ family, role inclusions had already been identified as a source of undecidability in expressive DLs because they loosely correspond to context-free grammars: if each inclusion $S_1 \cdots S_n \sqsubseteq S$ in a knowledge base is seen as a production rule $S \to S_1 \cdots S_n$, then the knowledge base induces a context-free language $\mathcal{L}(S)$ for each role S. Using this correspondence, Wessel (2001) showed that checking satisfiability of $\mathcal{ALCR}$ knowledge bases with unrestricted role inclusions is undecidable. To regain decidability, Horrocks and Sattler (2004) proposed a syntactic *regularity restriction* on role inclusions ensuring that each language $\mathcal{L}(S)$ is regular and can thus be recognised using a finite automaton (FA); Kazakov (2008) later showed that, in some cases, the size of this automaton is necessarily exponential in the knowledge base size. The OWL 2 DL profile of OWL 2 extends $\mathcal{ALCR}$ and thus incorporates the regularity restriction into its definition.

Even with unrestricted role inclusions, all standard reasoning problems for $\mathcal{EL}$ can be solved in polynomial time (Baader et al., 2005). Moreover, Stefanoni, Motik, and Horrocks (2013) showed that answering CQs over OWL 2 EL knowledge bases without role inclusions is NP-complete. However, using the correspondence between role inclusions and context-free grammars, Rosati (2007) and Krötzsch, Rudolph, and Hitzler (2007) independently proved that answering CQs over $\mathcal{EL}$ knowledge bases with unrestricted role inclusions is undecidable; furthermore, Krötzsch et al. (2007) also showed that checking concept subsumptions over $\mathcal{EL}$ knowledge bases with inverse roles and unrestricted role inclusions is undecidable.

OWL 2 EL inherits the regularity restriction from OWL 2 DL, and so the undecidability proofs by Rosati (2007) and Krötzsch et al. (2007) do not apply to OWL 2 EL. In fact, Krötzsch et al. (2007) showed that answering CQs over $\mathcal{EL}$ knowledge bases extended with regular role inclusions is PSpace-hard in combined complexity, and they proposed a CQ answering algorithm for a fragment of OWL 2 EL with regular role inclusions. This algorithm, however, runs in PSpace only if, for each role S, language $\mathcal{L}(S)$ can be represented using an automaton of polynomial size; due to the mentioned result by Kazakov (2008), this approach does not provide us with a matching PSpace upper bound for the problem. Ortiz et al. (2011) proposed a different algorithm for answering CQs over OWL 2 EL knowledge bases (with regular role inclusions and without any restriction on the usage of other features). Similarly to the algorithm by Krötzsch et al. (2007), the algorithm by Ortiz et al. (2011) also encodes regular role inclusions using finite automata. Hence, while both of these algorithms run in time polynomial in the size of the data and thus settle the question of data complexity, they do not settle the question of combined complexity.

There are comparatively few works on studying the complexity of (conjunctive) graph XPath queries over DL knowledge bases. In particular, Kostylev et al. (2014) observed that GXQs are closely related to *propositional dynamic logic with full negation* (Harel, Tiuryn, & Kozen, 2000), which immediately shows that answering GXQs over DL knowledge bases is undecidable even with respect to the empty knowledge base. Several GXQ fragments were proposed as a possible solution to this problem: *path-positive GXQs* disallow negation over role expressions, and *positive GXQs* further prohibit negation over concepts as well. Kostylev et al. (2014) showed that answering path-positive GXQs is intractable in data complexity already for queries without the transitive closure operator and for knowledge bases containing only instance assertions. Recently, Bienvenu et al. (2014) showed that answering positive GXQs in a fragment of OWL 2 EL is tractable in data complexity, but ExpTime-complete in combined complexity.

1.2 Our Contributions

In this paper, we present several novel complexity results on answering queries over OWL 2 EL knowledge bases.

First, we present the first CQ answering algorithm that can handle all of OWL 2 EL (with regular role inclusions but without any restriction on the size of the FAs) and that runs in PSpace, and thus we settle the open question of the combined complexity of CQ answering for OWL 2 EL. Our result is based on a novel encoding of the languages induced by regular role inclusions using pushdown automata (PDAs)—that is, FAs extended with a stack. We show that, for each role S, we can construct in polynomial time a PDA that accepts language $\mathcal{L}(S)$ and whose computations use a stack of size linear in the number of role inclusions. Bounded-stack PDAs (Anselmo, Giammarresi, & Varricchio, 2003) recognise precisely the class of regular languages and can be exponentially more succinct than finite automata (Geffert, Mereghetti, & Palano, 2010). To obtain a CQ answering algorithm running in PSpace, we extend the algorithm by Krötzsch et al. (2007)

	$\mathcal{ELHO}_{\bot}^{dr}$	OWL 2 EL	Horn-$\mathcal{SHOIQ}$	Horn-$\mathcal{SROIQ}$
data	PTime (Ortiz et al., 2011)	PTime (Theorem 31)	PTime (Ortiz et al., 2011)	PTime (Ortiz et al., 2011)
combined	NP (Stefanoni et al., 2013)	PSpace (Theorem 31)	ExpTime (Ortiz et al., 2011)	2ExpTime (Ortiz et al., 2011)

Table 1: The complexity landscape of CQ answering (all are completeness results)

to handle the *universal role*, *keys*, *self-restrictions*, and *reflexive roles*, thus covering all features of the EL profile apart from *datatypes*, and we adapt it so that it can handle regular role inclusions encoded using PDAs. Apart from allowing us to obtain the complexity results presented in this paper, the tableau algorithm by Horrocks, Kutz, and Sattler (2006) used in popular reasoners such as Pellet (Sirin, Parsia, Cuenca Grau, Kalyanpur, & Katz, 2007) and FaCT++ (Tsarkov & Horrocks, 2006) can be straightforwardly modified to use bounded-stack PDAs instead of FAs, which could eliminate a potential source of inefficiency in practice. Finally, for brevity and simplicity we do not deal with datatypes in this paper; however, the set of OWL 2 EL datatypes has been designed so as to enable datatype reasoning using an external datatype checking procedure (Baader, Brandt, & Lutz, 2008; Cuenca Grau et al., 2008) that can be easily incorporated into our algorithm.

Second, we improve the PSpace lower bound by Krötzsch et al. (2007) by showing that answering CQs in OWL 2 EL is PSpace-hard already if just the role inclusions are considered as part of the input (i.e., the conjunctive query, the TBox, and the ABox are all fixed). Furthermore, we show that CQs can be answered in polynomial time if the query and the role inclusions are fixed, which emphasises the observation that role inclusions are the main source of the problem's PSpace-hardness.

Third, we show that *positive, converse-free CGXQs*—that is, CGXQs that do not allow for negation over paths, negation of concepts, and the converse operator—can be answered over OWL 2 EL knowledge bases using polynomial space. In particular, OWL 2 EL allows for role inclusions, self-restrictions, and reflexive roles, which allow us to polynomially reduce answering a CGXQ to answering a CQ over an extended knowledge base. We also show that answering positive, converse-free GXQs (i.e., CGXQs with a single atom) can be done in time polynomial in the input size. This result is interesting because Bienvenu et al. (2014) proved that answering positive GXQs over $\mathcal{EL}$ knowledge bases is ExpTime-complete; hence, adding the converse operator increases the complexity of GXQs. Our results thus show that answering GXQs and CGXQs is as difficult as instance checking and answering conjunctive queries, respectively, which at least from a theoretical perspective makes GXQs and CGXQs appealing as query languages for OWL 2 EL knowledge bases.

1.3 Summary of the Complexity Landscape

Table 1 summarises the complexity landscape of answering CQs in various DLs related to OWL 2 EL. Here, $\mathcal{ELHO}_{\bot}^{dr}$ is the fragment of OWL 2 EL obtained by allowing only simple role inclusions of the form $T \sqsubseteq S$, and by disallowing the universal role, reflexive roles, self-restrictions, and datatypes, and the combined complexity result for this logic is due to Stefanoni et al. (2013). Furthermore, Horn-$\mathcal{SHOIQ}$ extends $\mathcal{ELHO}_{\bot}^{dr}$ with inverse roles and Horn qualified number restrictions, and Horn-$\mathcal{SROIQ}$ extends Horn-$\mathcal{SHOIQ}$ with role

	positive converse-free GXQs	positive converse-free CGXQs	positive GXQs	path-positive GXQs	GXQs
data	PTime-c (Theorem 34)	PTime-c (Theorem 34)	PTime-h (Bienvenu et al., 2014)	coNP-h (Kostylev et al., 2014)	coNP-h (Kostylev et al., 2014)
combined	PTime-c (Theorem 34)	PSpace-c (Theorem 34)	ExpTime-h (Bienvenu et al., 2014)	ExpTime-h (Bienvenu et al., 2014)	undecidable (Kostylev et al., 2014)

Table 2: The complexity of answering navigational queries over OWL 2 EL knowledge bases ('c' means 'complete', and 'h' means 'hard')

inclusions; the results for these logics are due to Ortiz et al. (2011). CQ answering is PTime-complete in data complexity in all cases, which is essentially due to the fact that all of these logics are Horn so no disjunctive reasoning is needed. For the combined complexity, the table illustrates how the presence of different constructs affects the complexity of answering CQs. In particular, extending $\mathcal{ELHO}_{\bot}^{dr}$ with role inclusions increases the complexity from NP to PSpace; by our PSpace lower bound, this increase is solely due to role inclusions. Furthermore, extending $\mathcal{ELHO}_{\bot}^{dr}$ with inverse roles increases the complexity from NP to ExpTime. Finally, extending OWL 2 EL with inverse roles increases the complexity from PSpace to 2ExpTime.

Table 2 summarises the complexity landscape of answering navigational queries over OWL 2 EL knowledge bases. As one can see, adding the converse operator increases the combined complexity of GXQs to ExpTime (Bienvenu et al., 2014). Moreover, adding negation over node tests increases the data complexity of GXQs to coNP, whereas adding negation over path expressions leads to the undecidability in combined complexity (Kostylev et al., 2014). In contrast, existential quantification over paths does not increase the complexity: answering positive, converse-free (C)GXQs over OWL 2 EL knowledge bases is as difficult as answering (C)RPQs over $\mathcal{EL}$ knowledge bases (Bienvenu et al., 2013).

1.4 Organisation of the Article

The rest of this article is organised as follows. In Section 2, we present the basic definitions of finite automata, pushdown automata, the DL underpinning OWL 2 EL, and conjunctive queries. In Section 3, we introduce our novel encoding of regular role inclusions using PDAs of bounded stack size. In Section 4, we present the CQ answering algorithm for OWL 2 EL and discuss its complexity. In Section 5, we present our improved PSpace lower-bound of answering CQs in OWL 2 EL. Finally, in Section 6, we introduce (conjunctive) graph XPath queries, we show how to reduce the problem of answering positive, converse-free conjunctive graph XPath queries to answering ordinary conjunctive queries, and we present the aforementioned complexity results.

2. Preliminaries

In this section we recapitulate the basic definitions of finite automata, pushdown automata, the DL $\mathcal{ELRO}^+$ underpinning OWL 2 EL, and conjunctive queries. In the rest of the paper, $[i..j]$ is the set containing each natural number $k \in \mathbb{N}$ such that $i \leq k \leq j$.

2.1 Automata and Language Theory

In this article, we use the standard notions of *alphabets* (which must be finite), *strings*, *string concatenation*, *Kleene operators*, and *languages* from formal language theory (Hopcroft, Motwani, & Ullman, 2003). We assume that alphabets do not contain the special symbol ε, which we will use to label transitions in automata that do not consume input symbols. Furthermore, ϵ is the empty word. Finally, for w and w' words, $|w|$ is the number of symbols occurring in w; and $w - w'$ is the unique word w'' such that $w := w'' \cdot w'$ if such w'' exists, and otherwise $w - w'$ is undefined.

2.1.1 Finite Automata

A *finite automaton* (FA) is a tuple $\mathcal{F} = \langle Q, \Sigma, \delta, i, f \rangle$ where Q is a finite set of states, Σ is the *input alphabet*, $\delta : Q \times \Sigma \cup \{\varepsilon\} \mapsto 2^Q$ is the *transition function*, $i \in Q$ is the *start state*, and $f \in Q$ is the *final state*. Such $\mathcal{F}$ is *deterministic* if $|\delta(s, \varepsilon)| = 0$ and $|\delta(s, c)| \leq 1$ for each $s \in Q$ and each $c \in \Sigma$; otherwise, $\mathcal{F}$ is nondeterministic. The *size* $|\mathcal{F}|$ of $\mathcal{F}$ is the number of symbols used to encode $\mathcal{F}$ on a tape of a Turing machine.

An *instantaneous description* of $\mathcal{F}$ is a pair $\langle s, w \rangle$ such that $s \in Q$ and $w \in \Sigma^*$. The *derivation relation* $\vdash$ *for* $\mathcal{F}$ is the smallest set such that, for all states s and s' in Q, each symbol $c \in \Sigma$, and each word $w \in \Sigma^*$, we have

- if $s' \in \delta(s, c)$, then $\langle s, c \cdot w \rangle \vdash \langle s', w \rangle$; and
- if $s' \in \delta(s, \varepsilon)$, then $\langle s, w \rangle \vdash \langle s', w \rangle$.

Let $\vdash^*$ be the reflexive and transitive closure of $\vdash$. Then, the *language accepted by* $\mathcal{F}$ is defined as $\mathcal{L}(\mathcal{F}) = \{w \in \Sigma^* \mid \langle i, w \rangle \vdash^* \langle f, \epsilon \rangle\}$. A language $\mathcal{L}$ is *regular* if and only if an FA $\mathcal{F}$ exists such that $\mathcal{L} = \mathcal{L}(\mathcal{F})$.

2.1.2 Pushdown Automata

A *pushdown automaton* (PDA) is a tuple $\mathcal{P} = \langle Q, \Sigma, \Gamma, \delta, i, I, f, F \rangle$ where Q is a finite set of *states*; Σ is the *input alphabet*; Γ is the *stack alphabet*; δ is a *transition function* mapping each state $s \in Q$, each symbol $c \in \Sigma \cup \{\varepsilon\}$, and each stack symbol $X \in \Gamma$ to a finite subset $\delta(s, c, X) \subseteq Q \times \Gamma^*$; $i \in Q$ is the *start state*; $I \in \Gamma^*$ is the *start stack*; $f \in Q$ is the *final state*; and $F \in \Gamma^*$ is the *final stack*. The *size* $|\mathcal{P}|$ of $\mathcal{P}$ is the number of symbols used to encode $\mathcal{P}$ on a tape of a Turing machine.

An *instantaneous description* of $\mathcal{P}$ is a triple $\langle s, w, \gamma \rangle$ such that $s \in Q$, $w \in \Sigma^*$, and $\gamma \in \Gamma^*$. We read the stack content γ from left to right—that is, the leftmost symbol in γ is the top of the stack. The *derivation relation* $\vdash$ *for* $\mathcal{P}$ is the smallest set such that, for all states s and s' in Q, each symbol $c \in \Sigma$, each word $w \in \Sigma^*$, each stack symbol $X \in \Gamma$, and all words γ and γ' in Γ^*, we have

- $\langle s', \gamma' \rangle \in \delta(s, c, X)$ implies $\langle s, c \cdot w, X \cdot \gamma \rangle \vdash \langle s', w, \gamma' \cdot \gamma \rangle$; and
- $\langle s', \gamma' \rangle \in \delta(s, \varepsilon, X)$ implies $\langle s, w, X \cdot \gamma \rangle \vdash \langle s', w, \gamma' \cdot \gamma \rangle$.

Let $\vdash^*$ be the reflexive and transitive closure of relation $\vdash$. Then, the *language accepted by* $\mathcal{P}$ is defined as $\mathcal{L}(\mathcal{P}) = \{w \in \Sigma^* \mid \langle i, w, I \rangle \vdash^* \langle f, \epsilon, F \rangle\}$.

Our definitions of a PDA $\mathcal{P}$ and of a language $\mathcal{L}(\mathcal{P})$ are somewhat nonstandard: the literature typically considers a *Hopcroft PDA* (Hopcroft et al., 2003) $\mathcal{P}_h$ that differs from our definition in that it does not contain the final stack F and its initial stack I is a symbol from Γ (rather than a word over Γ); moreover, the language accepted by $\mathcal{P}_h$ is defined as $\mathcal{L}_h(\mathcal{P}_h) = \{w \in \Sigma^* \mid \exists \gamma \in \Gamma^* : \langle i, w, I \rangle \vdash^* \langle f, \epsilon, \gamma \rangle\}$. We show next that our definitions are equivalent to the standard definitions by Hopcroft et al. (2003).

Proposition 1. *The following two properties hold.*

(1) For each PDA $\mathcal{P}$, a Hopcroft PDA $\mathcal{P}_h$ exists such that $\mathcal{L}(\mathcal{P}) = \mathcal{L}_h(\mathcal{P}_h)$.

(2) For each Hopcroft PDA $\mathcal{P}_h$, a PDA $\mathcal{P}$ exists such that $\mathcal{L}_h(\mathcal{P}_h) = \mathcal{L}(\mathcal{P})$.

Proof (Sketch). We first prove property (1), after which we prove property (2).

(1) We show how to transform an arbitrary PDA $\mathcal{P}$ into a Hopcroft PDA $\mathcal{P}_h$ such that $\mathcal{L}(\mathcal{P}) = \mathcal{L}_h(\mathcal{P}_h)$. Such $\mathcal{P}_h$ uses a fresh initial state i' and fresh stack symbols Z_0 and $\perp$ not occurring in Γ. Symbol Z_0 is the start stack symbol of $\mathcal{P}_h$; furthermore, $\mathcal{P}_h$ has a new ε-transition that moves the PDA from state i' to the initial state i of $\mathcal{P}$ by replacing Z_0 with $I \cdot \perp$, where I is the start stack of $\mathcal{P}$. At this point, $\mathcal{P}_h$ simulates $\mathcal{P}$, always leaving $\perp$ at the bottom of the stack until it reaches the final state f of $\mathcal{P}$. Next, $\mathcal{P}_h$ uses fresh states $s_1, \ldots, s_{|F|}$ and fresh ε-transitions that move $\mathcal{P}_h$ from state f to $s_{|F|}$ by reading F from the stack. Finally, from $s_{|F|}$, PDA $\mathcal{P}_h$ ε-moves to a fresh final state f' if the top-most symbol on the stack is $\perp$, thus accepting the input whenever $\mathcal{P}$ reaches f with F on its stack. Automata $\mathcal{P}$ and $\mathcal{P}_h$ clearly accept the same languages.

(2) We show how to transform an arbitrary Hopcroft PDA $\mathcal{P}_h$ into a PDA $\mathcal{P}$ such that $\mathcal{L}_h(\mathcal{P}_h) = \mathcal{L}(\mathcal{P})$. PDA $\mathcal{P}$ uses a fresh stack symbol $\perp$, its initial stack is $I \cdot \perp$ where I is the initial stack symbol of $\mathcal{P}_h$, and its final stack is the empty word. Then $\mathcal{P}$ simulates $\mathcal{P}_h$, always leaving $\perp$ at the bottom of the stack until it reaches the final state f of $\mathcal{P}_h$. Next, $\mathcal{P}$ ε-moves to a fresh final state f' and pops the topmost symbol off the stack. At this point, the PDA takes further ε-transitions to empty its stack, eventually reaching its final state with the empty stack. Automata $\mathcal{P}$ and $\mathcal{P}_h$ clearly accept the same languages. □

For k a natural number, the *k-bounded language accepted by* $\mathcal{P}$ is the set $\mathcal{L}_k(\mathcal{P})$ containing each word $w \in \Sigma^*$ for which a derivation $\langle s_0, w_0, \gamma_0 \rangle \vdash \cdots \vdash \langle s_n, w_n, \gamma_n \rangle$ exists where

- s_0 and s_n are the start and the final state of $\mathcal{P}$, respectively;
- $w_0 = w$ and $w_n = \epsilon$;
- γ_0 and γ_n are the start and the final stack of $\mathcal{P}$, respectively; and
- $|\gamma_i| \leq k$ for each $i \in [0..n]$.

Then, $\mathcal{P}$ has a *k-bounded stack* if $\mathcal{L}(\mathcal{P}) = \mathcal{L}_k(\mathcal{P})$. As the stack of $\mathcal{P}$ is bounded by a constant, PDA $\mathcal{P}$ can be simulated by a finite automaton that encodes the stack contents using its states, and so $\mathcal{L}(\mathcal{P})$ is regular, but translating $\mathcal{P}$ into a finite automaton may require space exponential in k (Geffert et al., 2010). In contrast, the following proposition shows that there exists a PDA $\mathcal{P}_k$ such that $\mathcal{L}(\mathcal{P}_k) = \mathcal{L}_k(\mathcal{P})$ and the size of $\mathcal{P}_k$ is polynomial in the size of $\mathcal{P}$ and k.

Proposition 2. *For each PDA $\mathcal{P}$ and natural number k, one can compute in polynomial time a PDA $\mathcal{P}_k$ such that $\mathcal{L}(\mathcal{P}_k) = \mathcal{L}_k(\mathcal{P})$.*

Proof. Let $\mathcal{P} = \langle Q, \Sigma, \Gamma, \delta, i, I, f, F\rangle$ be a PDA and let $k \in \mathbb{N}$ be a natural number. Let $\mathcal{P}_k = \langle Q_k, \Sigma, \Gamma, \delta_k, i_k, I, f_k, F\rangle$ be the PDA defined by

- $Q_k = Q \times [0..k]$;
- transition function δ_k is the smallest function such that, for each $\ell \in [0..k]$, each symbol $c \in \Sigma \cup \{\varepsilon\}$, all states $s, s' \in Q$, and each word $\gamma \in \Gamma^*$ such that $\langle s', \gamma\rangle \in \delta(s, c, X)$ and $\ell + |\gamma| - 1 \leq k$, we have $\langle\langle s', \ell + |\gamma| - 1\rangle, \gamma\rangle \in \delta_k(\langle s, \ell\rangle, c, X)$; and
- $i_k = \langle i, |I|\rangle$ and $f_k = \langle f, |F|\rangle$.

Clearly, $\mathcal{P}_k$ can be computed in time polynomial in the size of $\mathcal{P}$ and k. Let $\vdash$ and $\vdash_k$ be the derivation relations for $\mathcal{P}$ and $\mathcal{P}_k$, respectively. By the definitions of δ_k and i_k, we have that $\langle\langle s, \ell\rangle, w, \gamma\rangle \vdash_k \langle\langle s', j\rangle, w', \gamma'\rangle$ if and only if $\langle s, w, \gamma\rangle \vdash \langle s', w', \gamma'\rangle$, $|\gamma| = \ell$ and $|\gamma'| = j$, and $\max(\ell, j) \leq k$. Thus, we have $\mathcal{L}_k(\mathcal{P}) = \mathcal{L}(\mathcal{P}_k)$, as required. □

2.2 Description Logic $\mathcal{ELRO}^+$ and Conjunctive Queries

The description logic $\mathcal{ELRO}^+$, underpinning OWL 2 EL, is defined w.r.t. a signature consisting of mutually disjoint and countably infinite alphabets C, R, and I of *atomic concepts*, *roles*, and *individuals*, respectively. We assume that $\{\top_c, \bot_c\} \subseteq \mathsf{C}$, where $\top_c$ is the *top concept* and $\bot_c$ is the *bottom concept*; similarly, we assume that $\{\top_r, \bot_r\} \subseteq \mathsf{R}$, where $\top_r$ is the *top role* (universal role) and $\bot_r$ is the *bottom role*. For each individual $a \in \mathsf{I}$, expression $\{a\}$ is a *nominal*—that is, a concept consisting precisely of individual a. Then, N is the set containing nominal $\{a\}$ for each individual $a \in \mathsf{I}$. We call each $B \in \mathsf{C} \cup \mathsf{N}$ a *basic concept*. A *role chain* ρ is a word over R; for $|\rho| = 0$, we call ρ the *empty role chain* and we write it as ϵ. *Concepts*, *TBox axioms*, *RBox axioms*, and *ABox axioms* are defined as specified in Table 3. An $\mathcal{ELRO}^+$ *TBox* $\mathcal{T}$ is a finite set of concept inclusions, range restrictions, and keys; and an $\mathcal{ELRO}^+$ *RBox* $\mathcal{R}$ is a finite set of role inclusions.

For $\mathcal{R}$ an $\mathcal{ELRO}^+$ RBox, let $\Sigma_{\mathcal{R}} := \{\top_r\} \cup \{S \in \mathsf{R} \mid S \text{ occurs in } \mathcal{R}\}$; furthermore, the *rewrite relation $\Rightarrow$ w.r.t. $\mathcal{R}$* is the smallest relation on role chains such that the following holds for all role chains ρ_1 and ρ_2.

- $\rho_1 \cdot S \cdot \rho_2 \Rightarrow \rho_1 \cdot \rho \cdot \rho_2$ for each axiom $\rho \sqsubseteq S \in \mathcal{R}$.
- $\rho_1 \cdot \top_r \cdot \rho_2 \Rightarrow \rho_1 \cdot \rho \cdot \rho_2$ for each role chain $\rho \in \Sigma^*_{\mathcal{R}}$.

Then $\Rightarrow^*$ is the reflexive–transitive closure of $\Rightarrow$. For S a role, $\mathcal{L}(S) := \{\rho \in \mathsf{R}^* \mid S \Rightarrow^* \rho\}$ is the *language induced by RBox $\mathcal{R}$*. A role S is *simple* in $\mathcal{R}$ if, for each role chain ρ with

	Syntax	Semantics
Concepts:		
top concept	$\top_c$	$\Delta^{\mathcal{I}}$
bottom concept	$\bot_c$	$\emptyset$
nominal	$\{a\}$	$\{a^{\mathcal{I}}\}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
self-restriction	$\exists S.\mathsf{Self}$	$\{x \in \Delta^{\mathcal{I}} \mid \langle x, x\rangle \in S^{\mathcal{I}}\}$
existential restriction	$\exists S.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in C^{\mathcal{I}} : \langle x, y\rangle \in S^{\mathcal{I}}\}$
Role chains:		
top role	$\top_r$	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
bottom role	$\bot_r$	$\emptyset$
empty role chain	ϵ	$\{\langle x, x\rangle \mid x \in \Delta^{\mathcal{I}}\}$
nonempty role chain	$S_1 \cdots S_n$	$S_1^{\mathcal{I}} \circ \cdots \circ S_n^{\mathcal{I}}$
TBox axioms:		
concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
range restriction	$\mathsf{range}(S, C)$	$S^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times C^{\mathcal{I}}$
key	$\mathsf{key}(C, S_1 \ldots S_n)$	For all $x, y, z_1, \ldots, z_n$ in $\Delta^{\mathcal{I}}$ such that individuals $a, b, c_1, \ldots, c_n$ in I exist with $x = a^{\mathcal{I}}$, $y = b^{\mathcal{I}}$, and $z_i = c_i^{\mathcal{I}}$ for $1 \leq i \leq n$, $x = y$ holds whenever $\{x, y\} \subseteq C^{\mathcal{I}}$ and $\{\langle x, z_i\rangle, \langle y, z_i\rangle\} \subseteq S_i^{\mathcal{I}}$ for $1 \leq i \leq n$.
RBox axioms:		
role inclusion	$\rho \sqsubseteq S$	$\rho^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
ABox axioms:		
concept assertion	$A(b)$	$b^{\mathcal{I}} \in S^{\mathcal{I}}$
role assertion	$S(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in S^{\mathcal{I}}$

Table 3: Interpreting $\mathcal{ELRO}^+$ concepts, roles, and axioms in an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$

$S \Rightarrow^* \rho$, we have $|\rho| \leq 1$. An $\mathcal{ELRO}^+$ *ABox* $\mathcal{A}$ is a finite set of concept and role assertions. Finally, an $\mathcal{ELRO}^+$ *knowledge base* (KB) is a tuple $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$ where $\mathcal{T}$ is an $\mathcal{ELRO}^+$ TBox, $\mathcal{R}$ is an $\mathcal{ELRO}^+$ RBox, and $\mathcal{A}$ is an $\mathcal{ELRO}^+$ ABox such that

- for each concept $\exists S.\mathsf{Self}$ occurring in $\mathcal{T}$, role S is simple in $\mathcal{R}$; and
- for each $S_1 \cdots S_n \sqsubseteq S \in \mathcal{R}$ and each $\mathsf{range}(S', C) \in \mathcal{T}$ such that $S' \Rightarrow^* S$, a role $S'_n \in \mathsf{R}$ exists such that $S'_n \Rightarrow^* S_n$ and $\mathsf{range}(S'_n, C) \in \mathcal{T}$.

Let $|\mathcal{T}|$, $|\mathcal{R}|$, and $|\mathcal{A}|$ be the numbers of symbols needed to encode $\mathcal{T}$, $\mathcal{R}$, and $\mathcal{A}$, respectively, on a tape of a Turing machine, and let $|\mathcal{K}| = |\mathcal{T}| + |\mathcal{R}| + |\mathcal{A}|$. Furthermore, for α a knowledge base, a TBox, or an ABox, we define

$$\mathsf{I}_\alpha := \{a \in \mathsf{I} \mid a \text{ occurs in } \alpha\}, \;\; \mathsf{N}_\alpha := \{\{a\} \mid a \in \mathsf{I}_\alpha\}, \text{ and } \mathsf{C}_\alpha := \{A \in \mathsf{C} \mid A \text{ occurs in } \alpha\}.$$

The semantics of $\mathcal{ELRO}^+$ is defined as follows. An *interpretation* is a tuple $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$ where $\Delta^\mathcal{I}$ is a nonempty set of domain elements, called the *domain of* $\mathcal{I}$, and $\cdot^\mathcal{I}$ is the *interpretation function* that maps each individual $a \in \mathsf{I}$ to a domain element $a^\mathcal{I} \in \Delta^\mathcal{I}$, each atomic concept $A \in \mathsf{C} \setminus \{\top_c, \bot_c\}$ to a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and each atomic role $S \in \mathsf{R} \setminus \{\top_r, \bot_r\}$ to a relation $S^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. Function $\cdot^\mathcal{I}$ is extended to concepts and role chains as shown in the upper part of Table 3, where $\circ$ denotes composition of binary relations. An interpretation $\mathcal{I}$ is a *model* of $\mathcal{K}$ if it satisfies all axioms occurring in $\mathcal{K}$ as shown at the bottom of Table 3. Moreover, $\mathcal{K}$ is *consistent* if a model of $\mathcal{K}$ exists; $\mathcal{K}$ is *inconsistent* if no model of $\mathcal{K}$ exists; and $\mathcal{K}$ *entails* a first-order sentence ϕ (resp. a concept inclusion $C \sqsubseteq D$ or a role inclusion $\rho \sqsubseteq S$), written $\mathcal{K} \models \phi$ (resp. $\mathcal{K} \models C \sqsubseteq D$ or $\mathcal{K} \models \rho \sqsubseteq S$), if $\mathcal{I} \models \phi$ (resp. $C^\mathcal{I} \subseteq D^\mathcal{I}$ or $\rho^\mathcal{I} \subseteq S^\mathcal{I}$) for each model $\mathcal{I}$ of $\mathcal{K}$. By the definition of $\mathcal{L}(S)$, we have that $\rho \in \mathcal{L}(S)$ implies $\mathcal{K} \models \rho \sqsubseteq S$. Knowledge base consistency, entailment of concept inclusions, and entailment of role inclusions can be decided in polynomial time (Krötzsch, 2011; Baader et al., 2005).

2.2.1 Conjunctive Queries

A *term* is an individual or a variable. An *atom* is an expression of the form $A(t)$ or $R(t', t)$ where A is an atomic concept, R is a role, and t' and t are terms. A *conjunctive query* (CQ) is a formula $q = \exists \vec{y}.\psi(\vec{x}, \vec{y})$ with ψ a conjunction of atoms over variables $\vec{x} \cup \vec{y}$. Variables $\vec{x}$ are the *answer variables* of q. When $\vec{x}$ is empty, we call $q = \exists \vec{y}.\psi(\vec{y})$ a *Boolean CQ* (BCQ).

A *substitution* σ is a partial mapping from variables to terms; and $\mathsf{dom}(\sigma)$ and $\mathsf{rng}(\sigma)$ are the domain and the range of σ, respectively. For α a conjunction of atoms, $\sigma(\alpha)$ is the result of applying substitution σ to the atoms in α. Then, $\sigma(q) = \exists \vec{z}.\sigma(\psi)$, where $\vec{z}$ contains (i) $\sigma(y)$ for each variable $y \in \vec{y}$ such that $\sigma(y)$ is a variable, and (ii) each variable $y \in \vec{y}$ such that $\sigma(y)$ is undefined. Our definition of $\sigma(q)$ is somewhat nonstandard because quantified variables can also be replaced: for example, given $q = \exists y_1, y_2, y_3.R(y_1, y_2) \wedge T(y_1, y_3)$ and $\sigma = \{y_2 \mapsto a,\, y_3 \mapsto z\}$, we have $\sigma(q) = \exists y_1, z.R(y_1, a) \wedge T(y_1, z)$.

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ be an $\mathcal{ELRO}^+$ knowledge base and let $q = \exists \vec{y}.\psi(\vec{x}, \vec{y})$ be a CQ. Then q *is over* $\mathcal{K}$ if q uses only the predicates and the individuals occurring in $\mathcal{K}$. A substitution π is a *candidate answer* for q over $\mathcal{K}$, if $\mathsf{dom}(\sigma) = \vec{x}$ and $\mathsf{rng}(\sigma) \subseteq \mathsf{I}_\mathcal{K}$, and such π is a *certain answer* to q over $\mathcal{K}$ if and only if $\mathcal{K} \models \pi(q)$. Answering q over $\mathcal{K}$ amounts to computing the set of all certain answers to q over $\mathcal{K}$. As stated, CQ answering is a function problem; thus in this article we study the complexity of the associated decision problem named *BCQ answering*, which is the problem of deciding, given a Boolean CQ q over $\mathcal{K}$, whether $\mathcal{K} \models q$. Please note that BCQ answering is equivalent to the *recognition problem* which decides, given a CQ q over $\mathcal{K}$ and a candidate answer π, whether π is a certain answer to q over $\mathcal{K}$.

Following Vardi (1982), *combined complexity* assumes that both q and $\mathcal{K}$ are part of the input, and *data complexity* assumes that only the ABox $\mathcal{A}$ is part of the input.

2.3 Ensuring Decidability of BCQ Answering via Regularity

Rosati (2007) and Krötzsch et al. (2007) independently showed that answering Boolean CQs over $\mathcal{ELRO}^+$ knowledge bases is undecidable. Intuitively, role inclusions can 'simulate' derivations in context-free languages; thus, a Boolean CQ can check whether two context-free languages have a non-empty intersection, which is a known undecidable problem (Hopcroft et al., 2003).

To regain decidability, we next recapitulate the definition of so-called regular RBoxes by Horrocks and Sattler (2004). Let $\mathcal{R}$ be an $\mathcal{ELRO}^+$ RBox and let $\prec$ be the smallest transitive relation on $\Sigma_\mathcal{R}$ such that, for each $\rho \cdot T \cdot \rho' \sqsubseteq S \in \mathcal{R}$ with $S \neq T$, we have $T \prec S$. Then, RBox $\mathcal{R}$ is *regular* if $\prec$ is irreflexive and each role inclusion $\rho \sqsubseteq S \in \mathcal{R}$ is of the form

(t1) $\epsilon \sqsubseteq S$,

(t2) $S \cdot S \sqsubseteq S$,

(t3) $S_1 \cdots S_n \cdot S \sqsubseteq S$ and $S_i \neq S$ for each $i \in [1..n]$,

(t4) $S_1 \cdots S_n \sqsubseteq S$ and $S_i \neq S$ for each $i \in [1..n]$, or

(t5) $S \cdot S_1 \cdots S_n \sqsubseteq S$ and $S_i \neq S$ for each $i \in [1..n]$.

By induction on $\prec$ we then define the *level* $\mathsf{lv}(S)$ of each role $S \in \Sigma_\mathcal{R}$ as follows: $\mathsf{lv}(S) = 0$ if no $T \in \Sigma_\mathcal{R}$ exists such that $T \prec S$; otherwise, $\mathsf{lv}(S) = 1 + \max\{\mathsf{lv}(T) \mid T \prec S\}$. Clearly, $\mathsf{lv}(S)$ can be computed in time polynomial in $|\mathcal{R}|$. In Section 4 we show that BCQ answering over $\mathcal{ELRO}^+$ KBs with regular RBoxes is in PSpace.

2.4 Normalising $\mathcal{ELRO}^+$ Knowledge Bases

For simplicity, in the rest of this paper we assume that each $\mathcal{ELRO}^+$ knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ is *normalised*, which is the case if the following properties hold.

(n1) We have $\mathsf{I}_\mathcal{K} \neq \emptyset$, and $\mathcal{K} \not\models \{a\} \sqsubseteq \{b\}$ for all $\{a, b\} \subseteq \mathsf{I}_\mathcal{K}$ with $a \neq b$.

(n2) Each axiom in $\mathcal{T}$ is of one of the following forms, for $A_{(i)}$ basic concepts and S a role.

$$A_1 \sqcap A_2 \sqsubseteq A_3 \quad A_1 \sqsubseteq \exists S.A_2 \quad \exists S.A_1 \sqsubseteq A_2 \quad A \sqsubseteq \exists S.\mathsf{Self} \quad \exists S.\mathsf{Self} \sqsubseteq A$$

(n3) Each axiom $\rho \sqsubseteq S \in \mathcal{R}$ is such that $|\rho| \leq 2$ and $S \neq \top_r$, and each role in $\mathcal{T} \cup \mathcal{A}$ also occurs in $\mathcal{R}$.

We next show that each knowledge base $\mathcal{K}$ can be normalised in polynomial time without affecting the regularity of the RBox component nor the answers to Boolean CQs.

Proposition 3. *For each $\mathcal{ELRO}^+$ knowledge base $\mathcal{K}$ with a regular RBox and each Boolean CQ q over $\mathcal{K}$, one can compute in polynomial time a normalised $\mathcal{ELRO}^+$ knowledge base $\mathcal{K}'$ and a Boolean CQ q' such that*

- *the RBox of $\mathcal{K}'$ is regular, and*
- *q' is over $\mathcal{K}'$, and $\mathcal{K} \models q$ if and only if $\mathcal{K}' \models q'$.*

Proof. Let $\mathcal{K}$ be an $\mathcal{ELRO}^+$ KB with regular RBox and let q be a Boolean CQ over $\mathcal{K}$.

We first satisfy property (n1). Let $\mathcal{K}_1$ be obtained from $\mathcal{K}$ by extending the ABox of $\mathcal{K}$ with assertion $\top_c(c)$ for c a fresh individual; clearly, $\mathcal{K}_1 \models q$ if and only if $\mathcal{K} \models q$. Next, let $\mathcal{K}_2$ and q' be obtained from $\mathcal{K}_1$ and q, respectively, by uniformly substituting each individual a with an arbitrary, but fixed, individual b such that $\mathcal{K}_1 \models \{a\} \sqsubseteq \{b\}$. Entailment of concept inclusions can be decided in polynomial time, so $\mathcal{K}_2$ and q' can be computed in polynomial time. Moreover, $\mathcal{K}_2$ and q' are obtained by replacing each individual a with an arbitrary, but fixed individual b such that $a^{\mathcal{I}} = b^{\mathcal{I}}$ for each model $\mathcal{I}$ of $\mathcal{K}_1$, so q' is over $\mathcal{K}_2$, and $\mathcal{K}_2 \models q'$ if and only if $\mathcal{K}_1 \models q$.

We next satisfy property (n2). Let $\mathcal{K}_3$ be the result of eliminating all keys from $\mathcal{K}_2$. As one can see from Table 3, keys can only derive axioms of the form $\{a\} \sqsubseteq \{b\}$; moreover, the effects of such conclusions have already been captured by $\mathcal{K}_2$, and so $\mathcal{K}_3 \models q'$ if and only if $\mathcal{K}_2 \models q'$. Next, we eliminate in polynomial time all range restrictions occurring in $\mathcal{K}$ by applying the syntactic transformation by Baader et al. (2008); let $\mathcal{K}_4$ be the resulting knowledge base. Since the definition of $\mathcal{ELRO}^+$ knowledge base carefully restricts the interactions between role inclusions and range restrictions, we have $\mathcal{K}_4 \models q'$ if and only if $\mathcal{K}_3 \models q'$ (Baader et al., 2008). Next, following Krötzsch (2011), we compute in polynomial time a knowledge base $\mathcal{K}_5$ that satisfies (n2) such that $\mathcal{K}_5 \models q'$ if and only if $\mathcal{K}_4 \models q'$.

We next satisfy property (n3). Let $\mathcal{K}_6$ be the result of exhaustively decomposing each role inclusion $\rho \sqsubseteq S$ of the form (t3)–(t5) with $|\rho| > 2$ occurring in $\mathcal{K}_5$ according to the following rewrite rules, where each occurrence of role S' is fresh.

$$\begin{array}{lrcl}
\text{(t3)} & S_1 \cdots S_n \cdot S \sqsubseteq S & \mapsto & \{S' \cdot S \sqsubseteq S, \quad S_1 \cdots S_n \sqsubseteq S'\} \\
\text{(t4)} & S_1 \cdots S_n \sqsubseteq S & \mapsto & \{S' \cdot S_n \sqsubseteq S, \quad S_1 \cdots S_{n-1} \sqsubseteq S'\} \\
\text{(t5)} & S \cdot S_1 \cdots S_n \sqsubseteq S & \mapsto & \{S \cdot S' \sqsubseteq S, \quad S_1 \cdots S_n \sqsubseteq S'\}
\end{array}$$

Only linearly many rewrite steps are required to satisfy (n3), and the resulting RBox is regular. Furthermore, each model of $\mathcal{K}_6$ is also a model of $\mathcal{K}_5$ and each model $\mathcal{I}$ of $\mathcal{K}_5$ can be expanded to a model $\mathcal{J}$ of $\mathcal{K}_6$ by interpreting each role S' occurring in $\mathcal{K}_6 \setminus \mathcal{K}_5$ as $(S')^{\mathcal{J}} = (\rho')^{\mathcal{J}}$, where ρ' is the unique role chain such that $\rho' \sqsubseteq S'$ occurs in $\mathcal{K}_6$. Thus, we have $\mathcal{K}_6 \models q'$ if and only if $\mathcal{K}_5 \models q'$. Next, let $\mathcal{K}_7$ be the result of removing each axiom $\rho \sqsubseteq \top_r$ in $\mathcal{K}_6$; all removed axioms are tautologies, so we have $\mathcal{K}_7 \models q'$ if and only if $\mathcal{K}_6 \models q'$. Finally, let $\mathcal{K}'$ be the result of adding axiom $\bot_r \sqsubseteq S$, for each role S that occurs in $\mathcal{K}_7$ but does not occur in its RBox component. The axioms in $\mathcal{K}' \setminus \mathcal{K}_7$ preserve regularity and are tautologies, so $\mathcal{K}' \models q'$ if and only if $\mathcal{K}_7 \models q'$, as required. □

3. Encoding Regular RBoxes Succinctly Using Bounded-Stack PDAs

Each reasoning algorithm for a DL with role inclusions known to us uses a step that checks whether $\rho \in \mathcal{L}(S)$ holds for an arbitrary role chain ρ and a role S. For example, to check whether $\mathcal{K} \models S(a, b)$ holds, an algorithm must ensure that, in each model of $\mathcal{K}$, a role chain $\rho \in \mathcal{L}(S)$ exists connecting the elements interpreting a and b. Although they characterise languages $\mathcal{L}(S)$, role inclusions do not lend themselves well to language recognition, so all algorithms known to us transform role inclusions into another, more manageable form. This is analogous to the fact that, while regular expressions characterise regular languages, the former are routinely transformed into FAs in order to facilitate language recognition.

Horrocks and Sattler (2004) showed that, for each regular RBox $\mathcal{R}$ and each role S occurring in $\mathcal{R}$, one can construct an FA $\mathcal{F}_S$ such that $\mathcal{L}(\mathcal{F}_S) = \mathcal{L}(S)$. These FAs are used in a tableau decision procedure for $\mathcal{SROIQ}$—the DL underpinning OWL 2 DL (Horrocks et al., 2006). Given a $\mathcal{SROIQ}$ knowledge base, the tableau procedure tries to construct

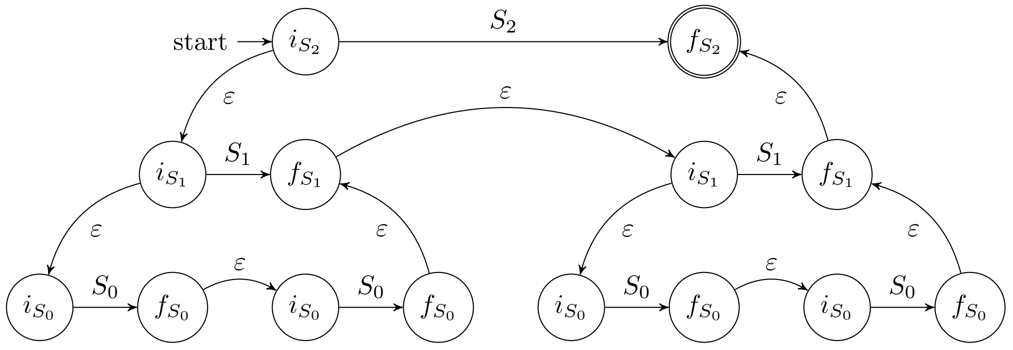

Figure 1: The FA $\mathcal{F}_{S_2}$ as constructed following Horrocks and Sattler (2004)

a finite graph representing a model of the KB, in which edges are labelled by roles, and vertices are labelled by concepts. The aforementioned FAs are used to ensure that universal restriction $\forall S.C$ obey the constraints imposed by role inclusions; roughly speaking, this is obtained by running $\mathcal{F}_S$ over the graph while updating the current state of $\mathcal{F}_S$ along the path, and by labelling each reachable vertex in which the state of $\mathcal{F}_S$ is final with concept C. Simančík (2012) optimised the tableau procedure by simulating FAs on-the-fly, rather than precomputing them in advance.

Horrocks and Sattler (2004) observed that their FAs can contain exponentially many states. Kazakov (2008) proved that this is unavoidable in some cases: for the regular RBox $\mathcal{R}_n$ containing axioms (1), the size of each FA $\mathcal{F}$ with $\mathcal{L}(\mathcal{F}) = \mathcal{L}(S_n)$ is exponential in n.

$$S_{i-1} \cdot S_{i-1} \sqsubseteq S_i \qquad\qquad \forall i \in [1..n] \tag{1}$$

This blowup in the number of states is caused by the simple model of computation underlying FAs, where the behaviour of the automaton is determined solely by the current state. In the example above, we have $\rho \in \mathcal{L}(S_n)$ whenever ρ consists of S_i repeated j times for some $i \in [0..n]$ with $j = 2^{n-i}$. Thus, while parsing such ρ, the FA recognising $\mathcal{L}(S_n)$ must 'remember' the number of occurrences of S_i it has already seen, which can be achieved only by using a different state for each number between 0 and 2^n. Figure 1 shows the FA $\mathcal{F}_{S_2}$ constructed by Horrocks and Sattler (2004): to 'remember' the current state, $\mathcal{F}_{S_2}$ contains two copies of automaton $\mathcal{F}_{S_1}$, and each copy of $\mathcal{F}_{S_1}$ contains two copies of automaton $\mathcal{F}_{S_0}$.

Hence, to obtain a PSpace procedure, we must devise a more succinct representation for the languages induced by role inclusions. Towards this goal, we note that role inclusions are closely related to context-free grammars, and that context-free languages can be efficiently recognised using pushdown automata (Hopcroft et al., 2003)—that is, FAs extended with an infinite stack for storing contextual information. Hence, given a regular RBox $\mathcal{R}$ and a role S occurring in $\mathcal{R}$, we construct a PDA $\mathcal{P}_S$ that accepts $\mathcal{L}(S)$. Unlike the FA shown in Figure 1 that 'remembers' contextual information using states, PDA $\mathcal{P}_S$ uses the stack to 'remember' the current status of the computation and determine how to proceed. We show that the number of states in $\mathcal{P}_S$ is polynomial in the size of $\mathcal{R}$, and that $\mathcal{P}_S$ can recognise $\mathcal{L}(S)$ by using a stack of size linear in the size of $\mathcal{R}$; thus, $\mathcal{P}_S$ provides us with the required succinct encoding of $\mathcal{F}_S$. In Section 4, we use these PDAs in an algorithm that answers Boolean CQs over $\mathcal{ELRO}^+$ knowledge bases using polynomial space.

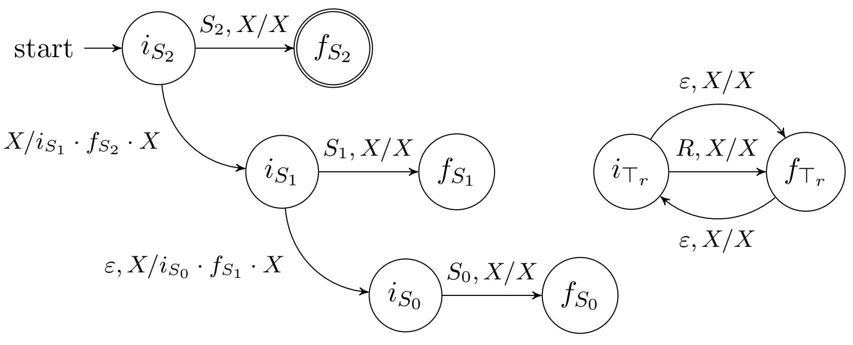

Figure 2: The PDA $\mathcal{P}_{S_2}$ corresponding to the FA $\mathcal{F}_{S_2}$, where $X \in \Gamma_{\mathcal{R}}$ and $R \in \Sigma_{\mathcal{R}}$

In the rest of this section, we fix an arbitrary regular RBox $\mathcal{R}$. By Proposition 3, we can assume that each role inclusion $\rho \sqsubseteq S \in \mathcal{R}$ is such that $|\rho| \leq 2$ and $S \neq \top_r$. For each role S occurring in $\Sigma_{\mathcal{R}}$, we next define the PDA $\mathcal{P}_S$.

Definition 4. *Let $S \in \Sigma_{\mathcal{R}}$ be a role. Then, $\mathcal{P}_S = \langle Q_{\mathcal{R}}, \Sigma_{\mathcal{R}}, \Gamma_{\mathcal{R}}, \delta_{\mathcal{R}}, i_S, \bot, f_S, \bot \rangle$ is the PDA where $Q_{\mathcal{R}} = \{i_T, f_T \mid T \in \Sigma_{\mathcal{R}}\}$ is the set of states, $\Gamma_{\mathcal{R}} = Q_{\mathcal{R}} \cup \{\bot\}$ is the stack alphabet, and $\delta_{\mathcal{R}}$ is the smallest transition function satisfying the following conditions for each $X \in \Gamma_{\mathcal{R}}$.*

(r) For each $T \in \Sigma_{\mathcal{R}} \setminus \{\top_r\}$, we have $\langle f_T, X \rangle \in \delta_{\mathcal{R}}(i_T, T, X)$.

(t1) For each $\epsilon \sqsubseteq T \in \mathcal{R}$, we have $\langle f_T, X \rangle \in \delta_{\mathcal{R}}(i_T, \varepsilon, X)$.

(t2) For each $T \cdot T \sqsubseteq T \in \mathcal{R}$, we have $\langle i_T, X \rangle \in \delta_{\mathcal{R}}(f_T, \varepsilon, X)$.

(t3) For each $T_1 \cdot T \sqsubseteq T \in \mathcal{R}$, we have $\langle i_{T_1}, i_T \cdot X \rangle \in \delta_{\mathcal{R}}(i_T, \varepsilon, X)$.

(t4) For each $T_1 \cdot T_2 \sqsubseteq T \in \mathcal{R}$, we have $\langle i_{T_1}, i_{T_2} \cdot f_T \cdot X \rangle \in \delta_{\mathcal{R}}(i_T, \varepsilon, X)$.

(t5) For each $T \cdot T_2 \sqsubseteq T \in \mathcal{R}$, we have $\langle i_{T_2}, f_T \cdot X \rangle \in \delta_{\mathcal{R}}(f_T, \varepsilon, X)$.

(ur) For each $T \in \Sigma_{\mathcal{R}}$, we have $\langle f_{\top_r}, X \rangle \in \delta_{\mathcal{R}}(i_{\top_r}, T, X)$.

(u1) $\langle f_{\top_r}, X \rangle \in \delta_{\mathcal{R}}(i_{\top_r}, \varepsilon, X)$.

(u2) $\langle i_{\top_r}, X \rangle \in \delta_{\mathcal{R}}(f_{\top_r}, \varepsilon, X)$.

(p) For each $T \in \Sigma_{\mathcal{R}}$ and each $s \in Q_{\mathcal{R}}$, we have $\langle s, \epsilon \rangle \in \delta_{\mathcal{R}}(f_T, \varepsilon, s)$.

In the following examples, we present the PDA that succinctly encodes the FA $\mathcal{F}_{S_2}$, and we explain the different types of transitions in Definition 4, and how the content of the stack influences the computation of PDAs.

Example 5. *Figure 2 shows the PDA $\mathcal{P}_{S_2}$ corresponding to the FA $\mathcal{F}_{S_2}$ in Figure 1. A transition $\langle s', \gamma \rangle \in \delta_{\mathcal{R}}(s, c, X)$ is shown as $s \xrightarrow{c, X/\gamma} s'$, where X/γ indicates that the transition replaces the top-most stack symbol X with word γ; moreover, transitions of the form (p) from Definition 4 are not shown in the figure for the sake of clarity. As one can see from the figure, unlike in FA $\mathcal{F}_{S_2}$, there is no copying of states in PDA $\mathcal{P}_{S_2}$.*

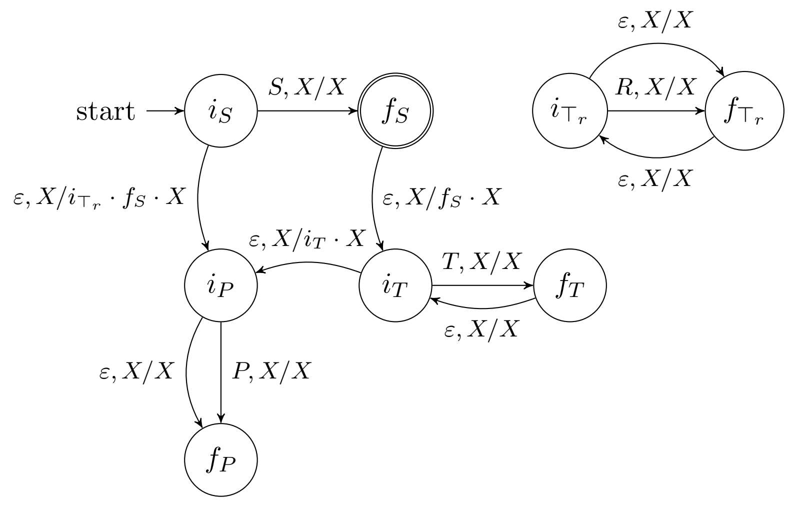

Figure 3: The PDA $\mathcal{P}_S$ for the RBox in Example 6, where $X \in \Gamma_{\mathcal{R}}$ and $R \in \Sigma_{\mathcal{R}}$

Example 6. *To explain the different types of transitions in Definition 4 and how the stack is used in the computation of a PDA, we use the regular RBox $\mathcal{R}$ containing role inclusions* (2)–(6)*. Figure 3 shows PDA $\mathcal{P}_S$ using the notation from Example 5.*

$$\epsilon \sqsubseteq P \tag{2}$$

$$T \cdot T \sqsubseteq T \tag{3}$$

$$P \cdot \top_r \sqsubseteq S \tag{4}$$

$$S \cdot T \sqsubseteq S \tag{5}$$

$$P \cdot T \sqsubseteq T \tag{6}$$

Each role $T \in \Sigma_{\mathcal{R}}$ is associated with states i_T and f_T, and moving from the former to the latter ensures that the PDA reads a role chain $\rho \in \mathcal{L}(T)$. A transition of type (r) allows the PDA to read T in state i_T. An ε-transition of type (t1) from i_T to f_T is added if T is reflexive, and it allows the PDA to read the empty role chain; in our example, axiom (2) *introduces the ε-transition from i_P to f_P. Moreover, an ε-transition of type (t2) from f_T to i_T is added if T is transitive, and it allows the PDA to read any number of role chains $\rho_1, \ldots, \rho_n \in \mathcal{L}(T)$; in our example, axiom* (3) *introduces the ε-transition from f_T to i_T. Transitions of types (ur), (u1), and (u2) analogously reflect the properties of $\top_r$: (ur) allows the PDA to read an arbitrary role, and (u1) and (u2) reflect the reflexivity and transitivity of $\top_r$, respectively. None of these transitions affect the PDA's stack.*

To illustrate transitions of type (t4), we next show how, for $\rho_1 = P \cdot S$, PDA $\mathcal{P}_S$ determines that $\rho_1 \in \mathcal{L}(S)$; the latter is ensured by axiom (4)*. Now assume that PDA $\mathcal{P}_S$ is in state i_S with $\perp$ on its stack. Due to axiom* (4)*, $\mathcal{P}_S$ can make an ε-transition of type (t4) to state i_P, pushing $i_{\top_r} \cdot f_S$ on the stack. Since the new state is i_P, the PDA will next need to read P; furthermore, the stack content signals to the PDA that, after it finishes reading P, it should move to state $i_{\top_r}$ to read $\top_r$ and then to state f_S to finish reading S. Indeed, $\mathcal{P}_S$ can then make a transition of type (r) to state f_P to read P, followed by an ε-transition of type (p) to state $i_{\top_r}$ popping $i_{\top_r}$ off the stack; next, the PDA can make a transition of* *type (ur) to state $f_{\top_r}$ reading S, followed by an ε-transition of type (p) to state f_S popping f_S off the stack. At this point, the PDA accepts the input.*

To illustrate transitions of types (t3) and (t5), we next show how, for $\rho_2 = S \cdot P \cdot T$, PDA $\mathcal{P}_S$ determines that $\rho_2 \in \mathcal{L}(S)$; the latter is ensured by axioms (5) *and* (6)*. Again, assume that PDA $\mathcal{P}_S$ is in state i_S with $\perp$ on its stack. PDA $\mathcal{P}_S$ can then make a transition of type (r) to state f_S, reading S and leaving the stack unchanged; next, due to axiom* (5)*, $\mathcal{P}_S$ can make an ε-transition of type (t5) to state i_T, pushing f_S on the stack. Due to axiom* (6)*, PDA $\mathcal{P}_S$ can next make an ε-transition of type (t3) to state i_P, pushing i_T on the stack; at this point, the stack contains $i_T \cdot f_S \cdot \perp$. Next, the PDA can make a transition of type (r) to state f_P reading P, and then an ε-transition of type (p) to state i_T popping i_T off the stack; furthermore, in an analogous way, the PDA can move to state f_T reading T and leaving $f_S \cdot \perp$ on the stack. Finally, the PDA can make an ε-transition of type (p) to state f_S popping f_S off the stack. At this point, the PDA accepts the input.*

To understand the benefit of using PDAs rather than FAs, note that $\mathcal{P}_S$ reaches state i_P while recognising both ρ_1 and ρ_2. Role P occurs in axioms (4) *and* (6)*, so when $\mathcal{P}_S$ moves into state i_P in order to read an occurrence of P, it must 'remember' which of the two axioms caused the move so that it knows how to continue after reading P: for ρ_1, $\mathcal{P}_S$ must continue reading $\top_r$, whereas for ρ_2, it must continue reading T. Unlike the FAs by Horrocks and Sattler (2004) that remember this information by copying states, $\mathcal{P}_S$ remembers this information on its stack: for ρ_1, it reaches i_P with $i_{\top_r} \cdot f_S \cdot \perp$ on its stack, whereas for ρ_2, $\mathcal{P}_S$ reaches i_P with $i_T \cdot f_S \cdot \perp$ on its stack. Thus, the stack of $\mathcal{P}_S$ is analogous to stacks in programming languages: stack symbols correspond to return addresses, and transitions of type (p) correspond to 'return' statements.*

The following proposition is immediate from the definition of PDA $\mathcal{P}_S$.

Proposition 7. *PDA $\mathcal{P}_S$ can be computed in time polynomial in $|\mathcal{R}|$.*

The following theorem states that PDA $\mathcal{P}_S$ accepts $\mathcal{L}(S)$ and that $\mathcal{P}_S$ has stack bounded by the size of $\mathcal{R}$. The proof of this result is given in Section 3.1.

Theorem 8. *For each role $S \in \Sigma_{\mathcal{R}}$ and each role chain ρ,*

1. *$\rho \in \mathcal{L}(\mathcal{P}_S)$ if and only if $\rho \in \mathcal{L}(S)$, and*
2. *$\mathcal{P}_S$ has stack bounded by $2 \cdot \mathsf{lv}(S) + 1$.*

Theorem 8 gives rise to the following notion of the depth of RBox $\mathcal{R}$, which provide us with a global bound on the stack size of the PDAs encoding $\mathcal{R}$.

Definition 9. *The* depth of the RBox $\mathcal{R}$ *is defined as* $\mathsf{d}_{\mathcal{R}} := \max_{S \in \Sigma_{\mathcal{R}}}(2 \cdot \mathsf{lv}(S) + 1)$.

Finally, we outline how our bounded-stack encoding of regular RBoxes can reduce the space used by the tableau algorithm for $\mathcal{SROIQ}$. Since $\mathcal{ELRO}^+$ does not support inverse roles, Definition 4 does not directly provide us with an encoding of the languages induced by $\mathcal{SROIQ}$ RBoxes. Nevertheless, we can extend the construction above by 'completing' RBox $\mathcal{R}$ so that $\mathsf{inv}(S_n) \cdots \mathsf{inv}(S_1) \sqsubseteq \mathsf{inv}(S) \in \mathcal{R}$ for each role inclusion $S_1 \cdots S_n \sqsubseteq S$ in the RBox, where $\mathsf{inv}(\cdot)$ maps each role to its inverse. One can check that, for each (inverse) role

S, the PDA $\mathcal{P}_S$ constructed using the completed RBox $\mathcal{R}$ encodes $\mathcal{F}_S$. Then, we can modify the portion of the tableau algorithm responsible for checking the satisfaction of universal restrictions by running a bounded-stack PDA over the graph constructed by the tableau procedure. Roughly speaking, for each universal restriction $\forall S.C$ labelling a vertex, we run $\mathcal{P}_S$ over the graph while updating the current state and the stack of $\mathcal{P}_S$, and we label each reachable vertex in which the current state and stack of $\mathcal{P}_S$ are final with concept C. Since $\mathcal{P}_S$ and its stack are of size polynomial in $|\mathcal{R}|$, this requires polynomial space, unlike the FAs by Horrocks and Sattler (2004) and the optimised encoding by Simančík (2012), which may require exponential space.

3.1 Proof of Correctness

In this section, we prove Theorem 8. Towards this goal, let $\vdash$ be the derivation relation w.r.t. transition function $\delta_\mathcal{R}$; furthermore, for each derivation step $\langle s, \rho, \gamma\rangle \vdash \langle s', \rho', \gamma'\rangle$, we write $\langle s, \rho, \gamma\rangle \vdash_x \langle s', \rho', \gamma'\rangle$ if $\langle s', \rho', \gamma'\rangle$ can be obtained from $\langle s, \rho, \gamma\rangle$ by applying a transition of the form (x) from Definition 4 with $x \in \{r, t1, \ldots, t5, ur, u1, u2, p\}$.

3.1.1 Soundness and Stack Boundedness

In this section, we prove that, for each role $S \in \Sigma_\mathcal{R}$ and each role chain ρ,

1. $\rho \in \mathcal{L}(\mathcal{P}_S)$ implies that $\rho \in \mathcal{L}(S)$, and
2. $\mathcal{P}_S$ has stack bounded by $2 \cdot \mathsf{lv}(S) + 1$.

To this end, we first show that PDA $\mathcal{P}_S$ satisfies the following *liveness property*: if during its computation $\mathcal{P}_S$ pushes a state $s \in Q_\mathcal{R}$ on the stack, then $\mathcal{P}_S$ will eventually pop s off the stack. Then, we show that each derivation of $\mathcal{P}_S$ moving from state i_S to state f_S takes one of five forms; we call such derivations *regular*. Finally, we show that regular derivations satisfy properties (1) and (2).

We start by showing that each PDA $\mathcal{P}_S$ satisfies the following liveness property.

Lemma 10. *Let $\langle s_0, \rho_0, \gamma_0 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_n, \rho_n, \gamma_n \cdot \gamma\rangle$ be an arbitrary derivation such that $s_0 = i_S$, $s_n = f_S$, and $\gamma_0 = \epsilon$ for some role $S \in \Sigma_\mathcal{R}$ and some word $\gamma \in \Gamma^*_\mathcal{R}$. Then, for each role T such that $\mathsf{lv}(T) < \mathsf{lv}(S)$ and each $i \in [0..n]$ such that $s_i \in \{i_T, f_T\}$ and $\gamma_i = s'_i \cdot \gamma'_i$ with $s'_i \in Q_\mathcal{R}$, an index $j \in [i..n]$ exists such that*

(a) $s_j = f_T$ and $\gamma_j = \gamma_i$;

(b) for each $k \in [i..j]$, word γ_k is of the form $\gamma_k := \gamma''_k \cdot \gamma_i$ for some $\gamma''_k \in \Gamma^*_\mathcal{R}$; and,

(c) $s_{j+1} = s'_i$, $\gamma_{j+1} = \gamma'_i$, and $\rho_{j+1} = \rho_j$.

Proof. Let $\langle s_0, \rho_0, \gamma_0 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_n, \rho_n, \gamma_n \cdot \gamma\rangle$ be as above, and for each $i \in [0..n-1]$, let $x_i \in \{r, t1, \ldots, t5, ur, u1, u2, p\}$ be the form of derivation step i—that is, we fix x_i (arbitrarily if there is more than one possibility) such that $\langle s_i, \rho_i, \gamma_i \cdot \gamma\rangle \vdash_{x_i} \langle s_{i+1}, \rho_{i+1}, \gamma_{i+1} \cdot \gamma\rangle$ holds. Furthermore, for each role T such that $\mathsf{lv}(T) < \mathsf{lv}(S)$, let I_T be the set containing each index $i \in [0..n]$ such that $s_i \in \{i_T, f_T\}$ and γ_i is of the form $\gamma_i := s'_i \cdot \gamma'_i$ with $s'_i \in Q_\mathcal{R}$. Note that, for each index $i \in I_T$, due to $\mathsf{lv}(T) < \mathsf{lv}(S)$, $s_i \in \{i_T, f_T\}$, and $s_n = f_S$, we have that $i < n$—that is, $\langle s_i, \rho_i, \gamma_i \cdot \gamma\rangle \vdash \langle s_{i+1}, \rho_{i+1}, \gamma_{i+1} \cdot \gamma\rangle$ occurs in our derivation. Next, by induction on $m \in \mathbb{N}$, we show that, for each role T with $m = \mathsf{lv}(T) < \mathsf{lv}(S)$ and each $i \in I_T$, some $j \in [i..n]$ exists satisfying properties (a)–(c).

Base case ($\diamondsuit$). Consider an arbitrary role $T \in \Sigma_{\mathcal{R}}$ such that $0 = \mathsf{lv}(T) < \mathsf{lv}(S)$. We consider the interesting case where $I_T \neq \emptyset$; otherwise, properties (a)–(c) hold vacuously. Since $\mathsf{lv}(T) = 0$ and $s_i \in \{i_T, f_T\}$, we have $x_i \in \{r, t1, t2, ur, u1, u2, p\}$. By reverse-induction on I_T (i.e., by induction starting from the maximal element), we next show that each index $i \in I_T$ satisfies the required properties.

Base case. Let $i = \max I_T$. If $x_i \in \{r, t1, t2, ur, u1, u2\}$, then $s_{i+1} \in \{i_T, f_T\}$ and $\gamma_{i+1} = \gamma_i$; thus, we have $i + 1 \in I_T$, which contradicts the maximality of i. The only remaining possibility is $x_i = p$, which implies that $s_i = f_T$, $s_{i+1} = s'_i$, $\gamma_{i+1} = \gamma'_i$, and $\rho_{i+1} = \rho_i$; but then, $j = i$ satisfies properties (a)–(c).

Inductive step. Consider an arbitrary index $i \in I_T$ such that properties (a)–(c) hold for each $\ell \in I_T$ with $\ell > i$. If $x_i \in \{r, t1, t2, ur, u1, u2\}$, then $s_{i+1} \in \{i_T, f_T\}$ and $\gamma_{i+1} = \gamma_i$; hence, $i_{i+1} \in I_T$ so, by the inductive hypothesis, an index j exists satisfying properties (a)–(c). Otherwise, if $x_i = p$, then $s_i = f_T$, $s_{i+1} = s'_i$, $\gamma_{i+1} = \gamma'_i$, and $\rho_{i+1} = \rho_i$, so $j = i$ satisfies properties (a)–(c).

Inductive Step ($\diamondsuit$). Consider an arbitrary $m \in \mathbb{N}$ such that properties (a)–(c) hold for each role $P \in \Sigma_{\mathcal{R}}$ with $\mathsf{lv}(P) \leq m$ and $\mathsf{lv}(P) < \mathsf{lv}(S)$ and each index in I_P. Furthermore, consider an arbitrary role T such that $m + 1 = \mathsf{lv}(T) < \mathsf{lv}(S)$. We consider the interesting case where $I_T \neq \emptyset$; otherwise, properties (a)–(c) hold vacuously. Recall that for each $\rho \sqsubseteq S' \in \mathcal{R}$ we have $S' \neq \top_r$, so $\mathsf{lv}(\top_r) = 0$ and $T \neq \top_r$. Thus, each $i \in I_T$ is such that $x_i \notin \{ur, u1, u2\}$. By reverse-induction on I_T, we next show that each index $i \in I_T$ satisfies the required properties.

Base case ($\heartsuit$). Let $i = \max I_T$. If $x_i \in \{r, t1, t2\}$, then $s_{i+1} \in \{i_T, f_T\}$ and $\gamma_{i+1} = \gamma_i$; thus, we have $i + 1 \in I_T$, which contradicts the maximality of i. If $x_i \in \{t3, t4, t5\}$, then $s_{i+1} \in \{i_P, f_P\}$ for some role P such that $\mathsf{lv}(P) < \mathsf{lv}(T)$ and $\mathsf{lv}(P) < \mathsf{lv}(S)$; furthermore, we have that γ_{i+1} is of the form $\gamma_{i+1} := \gamma''_{i+1} \cdot s_T \cdot \gamma_i$ where $s_T \in \{i_T, f_T\}$ and γ''_{i+1} is a sequence of zero or one states. Each state s occurring in γ''_{i+1} is such that $s \in \{i_R, f_R\}$ for some role R of level less than T. But then, by the inductive hypothesis ($\diamondsuit$), an index $\ell > i$ exists such that $s_\ell = s_T$ and $\gamma_\ell = \gamma_i$, which contradicts the maximality of i. Finally, if $x_i = p$, then $s_i = f_T$, $s_{i+1} = s'_i$, $\gamma_{i+1} = \gamma'_i$, and $\rho_{i+1} = \rho_i$, so $j = i$ satisfies properties (a)–(c).

Inductive step ($\heartsuit$). Consider an arbitrary index $i \in I_T$ such that properties (a)–(c) hold for each index $\ell \in I_T$ with $\ell > i$, and consider the possible forms of x_i.

- $x_i \in \{r, t1, t2\}$. Then, $s_{i+1} \in \{i_T, f_T\}$ and $\gamma_{i+1} = \gamma_i$, so $i + 1 \in I_T$. By the inductive hypothesis ($\heartsuit$), an index j exists satisfying properties (a)–(c).

- $x_i = t3$. Then, $s_{i+1} = i_{T_1}$ and $\gamma_{i+1} = i_T \cdot \gamma_i$ for some role T_1 with $\mathsf{lv}(T_1) < \mathsf{lv}(T)$. Thus, $i + 1 \in I_{T_1}$. By the inductive hypothesis ($\diamondsuit$), an index $\ell \in [i + 1..n]$ exists such that $s_\ell = f_{T_1}$ and $\gamma_\ell = \gamma_{i+1}$; furthermore, for each $k \in [i + 1..\ell]$, we have that γ_k is of the form $\gamma_k := \gamma''_k \cdot \gamma_{i+1}$ for some word $\gamma''_k \in \Gamma^*_{\mathcal{R}}$; finally, $s_{\ell+1} = i_T$ and $\gamma_{\ell+1} = \gamma_i$. By the definition of I_T, we have that $\ell + 1 \in I_T$. By the inductive hypothesis ($\heartsuit$), an index j exists satisfying properties (a)–(c).

- $x_i = t4$. Then, $s_{i+1} = i_{T_1}$ and $\gamma_{i+1} = i_{T_2} \cdot f_T \cdot \gamma_i$ for some roles T_1 and T_2 with $\mathsf{lv}(T_1) < \mathsf{lv}(T)$ and $\mathsf{lv}(T_2) < \mathsf{lv}(T)$. Thus, $i + 1 \in I_{T_1}$. By the inductive hypothesis ($\diamondsuit$), an index $\ell_1 \in [i+1..n]$ exists such that $s_{\ell_1} = f_{T_1}$ and $\gamma_{\ell_1} = \gamma_{i+1}$; furthermore, for each $k \in [i..\ell_1]$, we have that γ_k is of the form $\gamma_k := \gamma''_k \cdot \gamma_{i+1}$ for some word $\gamma''_k \in \Gamma^*_{\mathcal{R}}$; finally, $s_{\ell_1+1} = i_{T_2}$ and $\gamma_{\ell_1+1} = f_T \cdot \gamma_i$. Then, $\ell_1 + 1 \in I_{T_2}$. Again, by the inductive hypothesis ($\diamondsuit$), an index $\ell_2 \in [\ell_1 + 1..n]$ exists such that $s_{\ell_2} = f_{T_2}$ and $\gamma_{\ell_2} = \gamma_{\ell_1+1}$; furthermore, for each $k \in [\ell_1 + 1..\ell_2]$, we have that γ_k is of the form $\gamma_k := \gamma''_k \cdot \gamma_{\ell_1+1}$ for some word $\gamma''_k \in \Gamma^*_{\mathcal{R}}$; finally, $s_{\ell_2+1} = f_T$ and $\gamma_{\ell_2+1} = \gamma_i$. By the definition of I_T, we have that $\ell_2+1 \in I_T$. So, by the inductive hypothesis ($\heartsuit$), an index j exists satisfying properties (a)–(c).

- $x_i = t5$. Then, $s_{i+1} = i_{T_2}$ and $\gamma_{i+1} = f_T \cdot \gamma_i$ for some role T_2 with $\mathsf{lv}(T_2) < \mathsf{lv}(T)$. Then, $i + 1 \in I_{T_2}$. By the inductive hypothesis ($\diamondsuit$), an index $\ell \in [i + 1..n]$ exists such that $s_\ell = f_{T_2}$ and $\gamma_\ell = \gamma_{i+1}$; for each $k \in [i..\ell]$, we have that γ_k is of the form $\gamma_k := \gamma''_k \cdot \gamma_{i+1}$ for some word $\gamma''_k \in \Gamma^*_{\mathcal{R}}$; finally, $s_{\ell+1} = f_T$ and $\gamma_{\ell+1} = \gamma_i$. By the definition of I_T, we have that $\ell + 1 \in I_T$. So, by the inductive hypothesis ($\heartsuit$), an index j exists satisfying properties (a)–(c).

- $x_i = p$. Then, $s_i = f_T$, $s_{i+1} = s'_i$, $\gamma_{i+1} = \gamma'_i$, and $\rho_{i+1} = \rho_i$. Therefore, $j = i$ satisfies properties (a)–(c). □

Next, for each role $S \in \Sigma_{\mathcal{R}}$, we define the notion of *regular derivations of* $\mathcal{P}_S$.

Definition 11. *The set of regular derivations of $\mathcal{P}_{\top_r}$ is inductively defined as follows, for each role $T \in \Sigma_{\mathcal{R}}$, each role chain $\rho_i \in \mathsf{R}^*$, and each $\gamma \in \Gamma^*_{\mathcal{R}}$.*

seq_{ur} $\langle i_{\top_r}, T \cdot \rho_0, \gamma\rangle \vdash_{ur} \langle f_{\top_r}, \rho_0, \gamma\rangle$ *is a regular derivation of* $\mathcal{P}_{\top_r}$.

seq_{u1} $\langle i_{\top_r}, \rho_0, \gamma\rangle \vdash_{u1} \langle f_{\top_r}, \rho_0, \gamma\rangle$ *is a regular derivation of* $\mathcal{P}_{\top_r}$.

seq_{u2} *If* $\langle i_{\top_r}, \rho_0, \gamma\rangle \vdash \cdots \vdash \langle f_{\top_r}, \rho_k, \gamma\rangle$ *and* $\langle i_{\top_r}, \rho_k, \gamma\rangle \vdash \cdots \vdash \langle f_{\top_r}, \rho_n, \gamma\rangle$ *are regular derivations of* $\mathcal{P}_{\top_r}$*, then the following is also a regular derivation of* $\mathcal{P}_{\top_r}$.

$$\langle i_{\top_r}, \rho_0, \gamma\rangle \vdash \cdots \vdash \langle f_{\top_r}, \rho_k, \gamma\rangle \vdash_{u2} \langle i_{\top_r}, \rho_k, \gamma\rangle \vdash \cdots \vdash \langle f_{\top_r}, \rho_n, \gamma\rangle$$

Next, consider an arbitrary natural number $m \in \mathbb{N}$ and assume that regular derivations of $\mathcal{P}_T$ have already been defined for $T = \top_r$ and each role $T \in \Sigma_{\mathcal{R}} \setminus \{\top_r\}$ such that $\mathsf{lv}(T) \leq m$. Then, for each role $S \in \Sigma_{\mathcal{R}} \setminus \{\top_r\}$ with $\mathsf{lv}(S) = m+1$, regular derivations of $\mathcal{P}_S$ *are defined as follows, for each $S_{(i)} \in \Sigma_{\mathcal{R}}$, each $\rho_i \in \mathsf{R}^*$, and each $\gamma \in \Gamma^*_{\mathcal{R}}$.*

seq_r $\langle i_S, S \cdot \rho_0, \gamma\rangle \vdash_r \langle f_S, \rho_0, \gamma\rangle$ *is a regular derivation of* $\mathcal{P}_S$.

seq_{t1} *If* $\epsilon \sqsubseteq S \in \mathcal{R}$*, then* $\langle i_S, \rho_0, \gamma\rangle \vdash_{t1} \langle f_S, \rho_0, \gamma\rangle$ *is a regular derivation of* $\mathcal{P}_S$.

seq_{t2} *If* $S \cdot S \sqsubseteq S \in \mathcal{R}$ *and* $\langle i_S, \rho_0, \gamma\rangle \vdash \cdots \vdash \langle f_S, \rho_k, \gamma\rangle$ *and* $\langle i_S, \rho_k, \gamma\rangle \vdash \cdots \vdash \langle f_S, \rho_n, \gamma\rangle$ *are regular derivations of* $\mathcal{P}_S$*, then the following is also a regular derivation of* $\mathcal{P}_S$.

$$\langle i_S, \rho_0, \gamma\rangle \vdash \cdots \vdash \langle f_S, \rho_k, \gamma\rangle \vdash_{t2} \langle i_S, \rho_k, \gamma\rangle \vdash \cdots \vdash \langle f_S, \rho_n, \gamma\rangle$$

seq_{t3} *If $S_1 \cdot S \sqsubseteq S \in \mathcal{R}$, $\langle i_{S_1},\, \rho_0,\, i_S \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_1},\, \rho_k,\, i_S \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_1}$, and $\langle i_S,\, \rho_k,\, \gamma\rangle \vdash \cdots \vdash \langle f_S,\, \rho_n,\, \gamma\rangle$ is a regular derivation of $\mathcal{P}_S$, then the following is also a regular derivation of $\mathcal{P}_S$.*

$$\langle i_S,\, \rho_0,\, \gamma\rangle \vdash_{t3} \langle i_{S_1},\, \rho_0,\, i_S \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_1},\, \rho_k,\, i_S \cdot \gamma\rangle \vdash_p \langle i_S,\, \rho_k,\, \gamma\rangle \vdash \cdots \vdash \langle f_S,\, \rho_n,\, \gamma\rangle$$

seq_{t4} *If $S_1 \cdot S_2 \sqsubseteq S \in \mathcal{R}$, $\langle i_{S_1},\, \rho_0,\, i_{S_2} \cdot f_S \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_1},\, \rho_k,\, i_{S_2} \cdot f_S \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_1}$, and $\langle i_{S_2},\, \rho_k,\, f_S \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_2},\, \rho_n,\, f_S \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_2}$, then the following is also a regular derivation of $\mathcal{P}_S$.*

$$\begin{array}{llll}
\langle i_S,\ \rho_0,\ \gamma\rangle & \vdash_{t4} & & \\
\langle i_{S_1},\, \rho_0,\, i_{S_2} \cdot f_S \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_{S_1},\, \rho_k,\, i_{S_2} \cdot f_S \cdot \gamma\rangle & \vdash_p \\
\langle i_{S_2},\, \rho_k,\, f_S \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_{S_2},\, \rho_n,\, f_S \cdot \gamma\rangle & \vdash_p \\
\langle f_S,\ \rho_n,\ \gamma\rangle & & &
\end{array}$$

seq_{t5} *If $S \cdot S_2 \sqsubseteq S \in \mathcal{R}$, $\langle i_S,\, \rho_0,\, \gamma\rangle \vdash \cdots \vdash \langle f_S,\, \rho_k,\, \gamma\rangle$ is a regular derivation of $\mathcal{P}_S$, and $\langle i_{S_2},\, \rho_k,\, f_S \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_2},\, \rho_n,\, f_S \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_2}$, then the following is also a regular derivation of $\mathcal{P}_S$.*

$$\langle i_S,\, \rho_0,\, \gamma\rangle \vdash \cdots \vdash \langle f_S,\, \rho_k,\, \gamma\rangle \vdash_{t5} \langle i_{S_2},\, \rho_k,\, f_S \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_2},\, \rho_n,\, f_S \cdot \gamma\rangle \vdash_p \langle f_S,\, \rho_n,\, \gamma\rangle$$

We are left to show that each derivation of $\mathcal{P}_S$ that moves the PDA from the start state i_S to the final state f_S is regular and that regular derivations satisfy the required properties. In the following lemma, we show that derivations which leave a particular word γ at the bottom of the stack are regular and satisfy properties (1) and (2). Subsequently, we will show that each accepting derivation of $\mathcal{P}_S$ is of this form.

Lemma 12. *For each role $S \in \Sigma_{\mathcal{R}}$, each word $\gamma \in \Gamma^*_{\mathcal{R}}$, and each derivation of the form $\langle s_0,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_n,\, \rho_n,\, \gamma_n \cdot \gamma\rangle$ such that $s_0 = i_S$, $s_n = f_S$, and $\gamma_0 = \epsilon$,*

(i) the derivation is regular for $\mathcal{P}_S$;

(ii) for each $i \in [0..n]$, we have that $|\gamma_i| \leq 2 \cdot \mathsf{lv}(S)$; and

(iii) $S \Rightarrow^* \rho_0 - \rho_n$.

Proof. We prove the claim by induction on $n \in \mathbb{N}^+$.

Base case. For $n = 1$, consider an arbitrary role $S \in \Sigma_{\mathcal{R}}$, word $\gamma \in \Gamma^*_{\mathcal{R}}$, and sequence $\langle i_S,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash \langle f_S,\, \rho_1,\, \gamma_1 \cdot \gamma\rangle$. By Definition 4, only transitions from cases (r), (t1), (ur), and (u1) move $\mathcal{P}_S$ from state i_S to state f_S. These transitions leave the stack untouched, so $\gamma_1 = \epsilon = \gamma_0$ and property (*ii*) holds. For properties (*i*) and (*iii*), we next consider the four different forms that the sequence may take.

- $\langle i_S,\, S \cdot \rho_1,\, \gamma_0 \cdot \gamma\rangle \vdash_r \langle f_S,\, \rho_1,\, \gamma_1 \cdot \gamma\rangle$. Then $S \neq \top_r$, so this is a regular derivation of $\mathcal{P}_S$ by case seq_r and (*i*) holds. Finally, $\rho_0 - \rho_1 = S$, which implies $S \Rightarrow^* \rho_0 - \rho_1$, and so (*iii*) holds.

- $\langle i_S,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash_{t1} \langle f_S,\, \rho_0,\, \gamma_1 \cdot \gamma\rangle$. Then $S \neq \top_r$, so this is a regular derivation of $\mathcal{P}_S$ by case seq_{t1} and (*i*) holds. Finally, $\rho_0 - \rho_1 = \epsilon$; moreover, by case *t1* of Definition 4, we have $\epsilon \sqsubseteq S \in \mathcal{R}$, so $S \Rightarrow^* \epsilon$; hence, $S \Rightarrow^* \rho_0 - \rho_1$ and (*iii*) holds.

- $\langle i_S,\, T \cdot \rho_1,\, \gamma_0 \cdot \gamma\rangle \vdash_{ur} \langle f_S,\, \rho_1,\, \gamma_1 \cdot \gamma\rangle$. Then $S = \top_r$ and $T \in \Sigma_{\mathcal{R}}$, so this is a regular derivation of $\mathcal{P}_{\top_r}$ by case seq_{ur} and (*i*) holds. Finally, $\rho_0 - \rho_1 = T \in \Sigma_{\mathcal{R}}$, which implies $S \Rightarrow^* \rho_0 - \rho_1$, and so (*iii*) holds.

- $\langle i_S,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash_{u1} \langle f_S,\, \rho_0,\, \gamma_1 \cdot \gamma\rangle$. Then $S = \top_r$, so this is a regular derivation of $\mathcal{P}_{\top_r}$ by case seq_{u1} and (*i*) holds. Finally, $\rho_0 - \rho_1 = \epsilon$; hence, $S \Rightarrow^* \epsilon$, and so (*iii*) holds.

Inductive step. Consider an arbitrary $n \in \mathbb{N}^+$ and assume that *(i)–(iii)* hold for each role $S' \in \Sigma_{\mathcal{R}}$, each word $\gamma' \in \Gamma^*_{\mathcal{R}}$, and each derivation $\langle s'_0, \rho'_0, \gamma'_0 \cdot \gamma'\rangle \vdash \cdots \vdash \langle s'_c, \rho'_c, \gamma'_c \cdot \gamma'\rangle$ of length at most n and of the form required by this lemma. Furthermore, consider an arbitrary role $S \in \Sigma_{\mathcal{R}}$, an arbitrary word $\gamma \in \Gamma^*_{\mathcal{R}}$, and an arbitrary derivation

$$\langle s_0,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_{n+1},\, \rho_{n+1},\, \gamma_{n+1} \cdot \gamma\rangle \tag{7}$$

of length $n+1$ such that $s_0 = i_S$, $\gamma_0 = \epsilon$, and $s_{n+1} = f_S$. For each $i \in [0..n-1]$, let $x_i \in \{r, t1, \ldots, t5, ur, u1, u2, p\}$ be the form of derivation step i—that is, we fix x_i (arbitrarily if there is more than one possibility) such that $\langle s_i,\, \rho_i,\, \gamma_i \cdot \gamma\rangle \vdash_{x_i} \langle s_{i+1},\, \rho_{i+1},\, \gamma_{i+1} \cdot \gamma\rangle$ holds. We next consider the possible forms the sequence might have, and we show that properties (*i*)–(*iii*) hold in each case.

(*Case 1*) $S = \top_r$. We consider the form of $\langle s_0,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash_{x_0} \langle s_1,\, \rho_1,\, \gamma_1 \cdot \gamma\rangle$. Since $s_0 = i_{\top_r}$, we have $x_0 \in \{t1, t3, t4, ur, u1\}$. As $\mathcal{R}$ is normalised, each $\rho \sqsubseteq S' \in \mathcal{R}$ is such that $S' \neq \top_r$, so $x_0 \in \{ur, u1\}$ and we have $s_1 = f_{\top_r}$ and $\gamma_1 = \epsilon = \gamma_0$. Since $n > 1$, $\langle s_1, \rho_1, \gamma_1 \cdot \gamma\rangle \vdash_{x1} \langle s_2, \rho_1, \gamma_2 \cdot \gamma\rangle$ occurs in the sequence with $x_1 \in \{t2, t5, u2, p\}$. Since $s_1 = f_{\top_r}$ and $\mathcal{R}$ is normalised, we have $x_1 \in \{u2, p\}$; furthermore, since $\gamma_1 = \epsilon$ and by our assumption on the form of (7), we have $x_1 \neq p$. Hence, the only remaining possibility is that $x_1 = u2$. By case (u2) in Definition 4, we have $s_2 = i_{\top_r}$, $\rho_2 = \rho_1$, and $\gamma_2 = \gamma_1$. We next prove that properties (*i*)–(*iii*) hold.

(i) By seq_{ur} and seq_{u1}, $\langle s_0, \rho_0, \gamma_0 \cdot \gamma\rangle \vdash_{x_0} \langle s_1, \rho_1, \gamma_1 \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{\top_r}$. By the inductive hypothesis, $\langle s_2,\, \rho_2,\, \gamma_2 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_{n+1},\, \rho_{n+1},\, \gamma_{n+1} \cdot \gamma\rangle$ is also a regular derivation of $\mathcal{P}_S$. By the definition of regular derivations, we have $\gamma_n = \gamma_2 = \epsilon$. But then, (7) is a regular derivation of $\mathcal{P}_S$ by case seq_{u2}.

(ii) Words γ_0, γ_1, γ_2 are all empty. By the inductive hypothesis, we have $|\gamma_\ell| \leq 2 \cdot \mathsf{lv}(\top_r)$ for each $\ell \in [2..n+1]$. Therefore, $|\gamma_i| \leq 2 \cdot \mathsf{lv}(\top_r)$ holds for each $i \in [0..n+1]$.

(iii) By the inductive hypothesis, we have $\top_r \Rightarrow^* \rho_2 - \rho_{n+1}$. By cases (ur) and (u1), either $\rho_0 - \rho_2 = \epsilon$ or $\rho_0 - \rho_2 = T \in \Sigma_{\mathcal{R}}$. But then, $\top_r \Rightarrow^* \rho_0 - \rho_{n+1}$ holds.

(*Case 2*) $S \neq \top_r$ and $k \in [0..n]$ exists with $\langle s_k,\, \rho_k,\, \gamma_k \cdot \gamma\rangle \vdash_{t2} \langle s_{k+1},\, \rho_{k+1},\, \gamma_{k+1} \cdot \gamma\rangle$ and $s_k = f_S$. Then, by case (t2) in Definition 4, we have $S \cdot S \sqsubseteq S \in \mathcal{R}$, $s_{k+1} = i_S$, $\rho_{k+1} = \rho_k$, and $\gamma_{k+1} = \gamma_k$. We next prove that properties (*i*)–(*iii*) hold.

(i) By the inductive hypothesis, $\langle s_0,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_k,\, \rho_k,\, \gamma_k \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_S$. By the definition of regular derivations, we have $\gamma_k = \gamma_0 = \epsilon$. Since $s_{k+1} = i_S$ and $\gamma_{k+1} = \gamma_k = \epsilon$, we have that $\langle s_{k+1},\, \rho_{k+1},\, \gamma_{k+1} \cdot \gamma\rangle \vdash \cdots \vdash \langle s_{n+1},\, \rho_{n+1},\, \gamma_{n+1} \cdot \gamma\rangle$ is of the form shown in (7) and it is shorter than $n+1$ so, by the inductive hypothesis, it is a regular derivation of $\mathcal{P}_S$. Then, (7) is a regular derivation of $\mathcal{P}_S$ by case seq_{t2}.

(ii) By the inductive hypothesis, we have $|\gamma_{\ell_1}| \leq 2 \cdot \mathsf{lv}(S)$ for each $\ell_1 \in [0..k]$, as well as $|\gamma_{\ell_2}| \leq 2 \cdot \mathsf{lv}(S)$ for each $\ell_2 \in [k+1..n+1]$. Therefore, $|\gamma_i| \leq 2 \cdot \mathsf{lv}(S)$ holds for each $i \in [0..n+1]$.

(iii) By the inductive hypothesis, we have $S \Rightarrow^* \rho_0 - \rho_k$ and $S \Rightarrow^* \rho_{k+1} - \rho_{n+1}$. But then, $S \cdot S \sqsubseteq S \in \mathcal{R}$ and $\rho_{k+1} = \rho_k$ implies that $S \Rightarrow^* \rho_0 - \rho_{n+1}$ holds.

(*Case 3*) $S \neq \top_r$ and no $\ell \in [0..n]$ exists with $\langle s_\ell, \rho_\ell, \gamma_\ell \cdot \gamma\rangle \vdash_{t2} \langle s_{\ell+1}, \rho_{\ell+1}, \gamma_{\ell+1} \cdot \gamma\rangle$ and $s_\ell = f_S$, but $k \in [0..n]$ exists such that $\langle s_k, \rho_k, \gamma_k \cdot \gamma\rangle \vdash_{t5} \langle s_{k+1}, \rho_{k+1}, \gamma_{k+1} \cdot \gamma\rangle$ and $s_k = f_S$. Then, let k be the largest such index—that is, we assume that no $m > k$ exists such that $\langle s_m,\, \rho_m,\, \gamma_m \cdot \gamma\rangle \vdash_{t5} \langle s_{m+1},\, \rho_{m+1},\, \gamma_{m+1} \cdot \gamma\rangle$ and $s_m = f_S$. Then, by case (t5) in Definition 4, for some role S_2 of level less than S, we have that $S \cdot S_2 \sqsubseteq S \in \mathcal{R}$, $s_{k+1} = i_{S_2}$, $\rho_{k+1} = \rho_k$, and $\gamma_{k+1} = f_S \cdot \gamma_k$. We next prove that properties (*i*)–(*iii*) hold.

(i) Since $s_k = f_S$, by the inductive hypothesis then $\langle s_0,\, \rho_0,\, \gamma_0 \cdot \gamma\rangle \vdash \cdots \vdash \langle s_k,\, \rho_k,\, \gamma_k \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_S$. By Definition 12, we have that $\gamma_k = \gamma_0$. Since $s_{k+1} = i_{S_2}$ and $\gamma_{k+1} = f_S \cdot \gamma_0$, by Lemma 10, an index $j \in [k+1..n]$ exists such that $s_j = f_{S_2}$ and $\gamma_j = \gamma_{k+1}$; furthermore, $s_{j+1} = f_S$ and $\gamma_{j+1} = \gamma_0$ and $\rho_{j+1} = \rho_j$. We prove that $j+1 = n+1$. For the sake of contradiction, assume that $j+1 < n+1$ and consider the form of transition $\langle s_{j+1},\, \rho_{j+1},\, \gamma_{j+1} \cdot \gamma\rangle \vdash_{x_{j+1}} \langle s_{j+2},\, \rho_{j+2},\, \gamma_{j+2} \cdot \gamma\rangle$. Given that $s_{j+1} = f_S$ and $S \neq \top_r$, we must have $x_{j+1} \in \{t2, t5, p\}$. By the initial assumption, we have $x_{j+1} \neq t2$; furthermore, by the maximality of k, we have $x_{j+1} \neq t5$; finally, since $\gamma_{j+1} = \gamma_0 = \epsilon$, we have $x_{j+1} \neq p$. Thus, $j+1 = n+1$, as required. It follows that the sequence is of the following form, where $\rho_{k+1} = \rho_k$ and $\rho_{n+1} = \rho_n$.

$$\begin{array}{lllll}
\langle i_S,\ \rho_0, & \gamma_0 \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_S,\ \rho_k,\ \gamma_0 \cdot \gamma\rangle & \vdash_{t5} \\
\langle i_{S_2},\ \rho_{k+1}, & \gamma_{k+1} \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_{S_2},\ \rho_n,\ \gamma_n \cdot \gamma\rangle & \vdash_p \\
\langle f_S,\ \rho_{n+1}, & \gamma_0 \cdot \gamma\rangle & & &
\end{array}$$

By Lemma 10, for each $\ell \in [k+1..n]$, we have that γ_ℓ is of the form $\gamma_\ell = \gamma''_\ell \cdot f_S \cdot \gamma_0$. In particular, words γ''_{k+1} and γ''_n are both empty. Then, by the inductive hypothesis, we have that $\langle i_{S_2},\, \rho_{k+1},\, \gamma_{k+1} \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_2},\, \rho_n,\, \gamma_n \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_2}$. By case seq_{t5}, then (7) is a regular derivation of $\mathcal{P}_S$.

(ii) By the inductive hypothesis, for each $\ell_1 \in [0..k]$, we have that $|\gamma_{\ell_1}| \leq 2 \cdot \mathsf{lv}(S)$. Furthermore, for each $\ell_2 \in [k+1..n]$, we have that $|\gamma''_{\ell_2}| \leq 2 \cdot \mathsf{lv}(S_2)$. Since $\mathsf{lv}(S_2) < \mathsf{lv}(S)$ and $\gamma_{\ell_2} = \gamma''_{\ell_2} \cdot f_S$, we also have that $|\gamma_{\ell_2}| \leq 2 \cdot \mathsf{lv}(S)$. Given that $\gamma_{n+1} = \epsilon$, for each $i \in [0..n+1]$, we have that $|\gamma_i| \leq 2 \cdot \mathsf{lv}(S)$.

(iii) By the inductive hypothesis, we have that $S \Rightarrow^* \rho_0 - \rho_k$ and $S_2 \Rightarrow^* \rho_{k+1} - \rho_n$. Given that $S \Rightarrow^* S \cdot S_2$, that $\rho_{k+1} = \rho_k$, and that $\rho_{n+1} = \rho_n$, we obtain that $S \Rightarrow^* \rho_0 - \rho_{n+1}$.

(*Case 4*) $S \neq \top_r$ and no $\ell \in [0..n]$ exists with $\langle s_\ell, \rho_\ell, \gamma_\ell \cdot \gamma\rangle \vdash_{x_\ell} \langle s_{\ell+1}, \rho_{\ell+1}, \gamma_{\ell+1} \cdot \gamma\rangle$, $s_\ell = f_S$ and $x_\ell \in \{t2, t5\}$; but $\langle s_0, \rho_0, \gamma_0 \cdot \gamma\rangle \vdash_{t3} \langle s_1, \rho_1, \gamma_1 \cdot \gamma\rangle$. Then, by case (t3) in Definition 4, for some role S_1 of level less than S, we have that $S_1 \cdot S \sqsubseteq S \in \mathcal{R}$, $s_1 = i_{S_1}$, $\rho_1 = \rho_0$, and $\gamma_1 = i_S \cdot \gamma_0$. We next prove that properties (*i*)–(*iii*) hold.

(i) Since $s_1 = i_{S_1}$ and $\gamma_1 = i_S \cdot \gamma_0$, by Lemma 10, some $j \in [1..n]$ exists such that $s_j = f_{S_1}$ and $\gamma_j = \gamma_1$; furthermore, $s_{j+1} = i_S$ and $\gamma_{j+1} = \gamma_0$ and $\rho_{j+1} = \rho_j$. Then, the sequence is of the following form, where $\rho_1 = \rho_0$.

$$\begin{array}{llll} \langle i_S,\ \rho_0,\ \gamma_0 \cdot \gamma\rangle & \vdash_{t3} & & \\ \langle i_{S_1}, \rho_1,\ \gamma_1 \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_{S_1}, \rho_j,\ \gamma_j \cdot \gamma\rangle & \vdash_p \\ \langle i_S,\ \rho_{j+1}, \gamma_{j+1} \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_S,\ \rho_{n+1}, \gamma_{n+1} \cdot \gamma\rangle & \end{array}$$

By Lemma 10, for each $\ell \in [1..j]$, we have that γ_ℓ is of the form $\gamma_\ell = \gamma''_\ell \cdot i_S \cdot \gamma_0$. In particular, words γ''_1 and γ''_j are both empty. By the inductive hypothesis, then $\langle i_{S_1}, \rho_0, \gamma_1 \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_1}, \rho_j, \gamma_j \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_1}$. Since $\gamma_{j+1} = \gamma_0$, by the inductive hypothesis, then $\langle i_S, \rho_{j+1}, \gamma_{j+1} \cdot \gamma\rangle \vdash \cdots \vdash \langle f_S, \rho_{n+1}, \gamma_{n+1} \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_S$. By case seq_{t3}, then (7) is a regular derivation of $\mathcal{P}_S$.

(ii) By the inductive hypothesis, for each $\ell_2 \in [j+1..n+1]$, we have that $|\gamma_{\ell_2}| \leq 2 \cdot \mathsf{lv}(S)$; furthermore, for each $\ell_1 \in [1..j]$, we have that $|\gamma''_{\ell_1}| \leq 2 \cdot \mathsf{lv}(S_1)$. Since $\mathsf{lv}(S_1) < \mathsf{lv}(S)$ and $\gamma_{\ell_1} = \gamma''_{\ell_1} \cdot i_S$, we also have that $|\gamma_{\ell_1}| \leq 2 \cdot \mathsf{lv}(S)$. Finally, since $\gamma_0 = \epsilon$, for each $i \in [0..n+1]$, we have that $|\gamma_i| \leq 2 \cdot \mathsf{lv}(S)$.

(iii) By the inductive hypothesis, we have that $S_1 \Rightarrow^* \rho_1 - \rho_j$ and that $S \Rightarrow^* \rho_{j+1} - \rho_{n+1}$. Given that $S \Rightarrow^* S_1 \cdot S$, that $\rho_1 = \rho_0$, and that $\rho_{j+1} = \rho_j$, we have that $S \Rightarrow^* \rho_0 - \rho_{n+1}$.

(*Case 5*) $S \neq \top_r$ and no $\ell \in [0..n]$ exists with $\langle s_\ell, \rho_\ell, \gamma_\ell \cdot \gamma\rangle \vdash_{x_\ell} \langle s_{\ell+1}, \rho_{\ell+1}, \gamma_{\ell+1} \cdot \gamma\rangle$, $s_\ell = f_S$, and $x_\ell \in \{t2, t5\}$; in addition, $\langle s_0, \rho_0, \gamma_0 \cdot \gamma\rangle \vdash_{x_0} \langle s_1, \rho_1, \gamma_1 \cdot \gamma\rangle$ is such that $x_0 \neq t3$. We next consider the remaining possibilities for x_0. As $s_0 = i_S$, we have $x_0 \notin \{t2, t5, u2, p\}$ by cases (t2), (t5), (u2), and (p) of Definition 4; furthermore, due to $S \neq \top_r$, we have $x_0 \notin \{ur, u1\}$ by cases (ur) and (u1) of Definition 4. Moreover, assume that $x_0 \in \{r, t1\}$; then, we have $s_1 = f_S$ and $\gamma_1 = \gamma_0$ by cases (r) and (t1) of Definition 4; since $n > 1$ and $S \neq \top_r$, the only possibility is that $\langle s_1, \rho_1, \gamma_1 \cdot \gamma\rangle \vdash_p \langle s_2, \rho_2, \gamma_2 \cdot \gamma\rangle$, which is impossible due to $\gamma_1 = \epsilon$ and our assumption on the form of (7). Hence, the only remaining possibility is that $x_0 = t4$. By case (t4) in Definition 4, for some roles S_1 and S_2 of level less than S, we have $S_1 \cdot S_2 \sqsubseteq S \in \mathcal{R}$, $s_1 = i_{S_1}$, $\rho_1 = \rho_0$, and $\gamma_1 = i_{S_2} \cdot f_S \cdot \gamma_0$. We next prove that properties (*i*)–(*iii*) hold.

(i) Since $s_1 = i_{S_1}$ and $\gamma_1 = i_{S_2} \cdot f_S \cdot \gamma_0$, by Lemma 10, $j_1 \in [1..n]$ exists such that $s_{j_1} = f_{S_1}$ and $\gamma_{j_1} = \gamma_1$; furthermore, $s_{j_1+1} = i_{S_2}$ and $\gamma_{j_1+1} = f_S \cdot \gamma_0$ and $\rho_{j_1+1} = \rho_{j_1}$. Again, by Lemma 10, $j_2 \in [j_1+1..n]$ exists such that $s_{j_2} = f_{S_2}$ and $\gamma_{j_2} = \gamma_{j_1+1}$; furthermore, $s_{j_2+1} = f_S$ and $\gamma_{j_2+1} = \gamma_0$ and $\rho_{j_2+1} = \rho_{j_2}$. Next, we prove that $j_2 + 1 = n + 1$. For the sake of contradiction, suppose that $j_2 + 1 < n + 1$ and consider the form of $\langle s_{j_2+1}, \rho_{j_2+1}, \gamma_{j_2+1} \cdot \gamma\rangle \vdash_{x_{j_2+1}} \langle s_{j_2+2}, \rho_{j_2+2}, \gamma_{j_2+2} \cdot \gamma\rangle$. Given that $s_{j_2+1} = f_S$, we must have that $x_{j_2+1} \in \{t2, t5, u2, p\}$. However, we assumed that $x_{j_2+1} \notin \{t2, t5\}$ and that $S \neq \top_r$, so $x_{j_2+1} \neq u2$; finally, since $\gamma_{j_2+1} = \gamma_0 = \epsilon$, we have $x_{j_2+1} \neq p$.

Therefore, we have $j_2 + 1 = n + 1$, as required, and the sequence is of the following form, for $\rho_1 = \rho_0$, $\rho_{j_1+1} = \rho_j$, and $\rho_{n+1} = \rho_n$.

$$\begin{array}{llll}
\langle i_S,\ \rho_0, & \gamma_0 \cdot \gamma\rangle & \vdash_{t4} & \\
\langle i_{S_1}, \rho_1, & \gamma_1 \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_{S_1}, \rho_{j_1}, \gamma_{j_1} \cdot \gamma\rangle \ \vdash_p \\
\langle i_{S_2}, \rho_{j_1+1}, & \gamma_{j_1+1} \cdot \gamma\rangle & \vdash \cdots \vdash & \langle f_{S_2}, \rho_n,\ \gamma_n \cdot \gamma\rangle \ \vdash_p \\
\langle f_S,\ \rho_{n+1}, & \gamma_{n+1} \cdot \gamma\rangle & &
\end{array}$$

By Lemma 10, for each $\ell_1 \in [1..j_1]$, word γ_{ℓ_1} is of the form $\gamma_{\ell_1} = \gamma''_{\ell_1} \cdot i_{S_2} \cdot f_S \cdot \gamma_0$. In particular, words γ''_1 and γ''_{j_1} are both empty. Then, by the inductive hypothesis, we have that $\langle i_{S_1}, \rho_1, \gamma_1 \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_1}, \rho_{j_1}, \gamma_{j_1} \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_1}$. Similarly, by Lemma 10, for each $\ell_2 \in [j_1 + 1..n]$, we have that γ_{ℓ_2} is of the form $\gamma_{\ell_2} = \gamma''_{\ell_2} \cdot f_S \cdot \gamma_0$. In particular, words γ''_{j_1+1} and γ''_n are both empty. Then, by the inductive hypothesis, we have that $\langle i_{S_2}, \rho_{j_1+1}, \gamma_{j_1+1} \cdot \gamma\rangle \vdash \cdots \vdash \langle f_{S_2}, \rho_n, \gamma_n \cdot \gamma\rangle$ is a regular derivation of $\mathcal{P}_{S_2}$. By case seq_{t4}, then (7) is a regular derivation of $\mathcal{P}_S$.

(ii) By the inductive hypothesis, for each $\ell_1 \in [1..j_1]$, we have that $|\gamma''_{\ell_1}| \leq 2 \cdot \mathsf{lv}(S_1)$. Since $\mathsf{lv}(S_1) < \mathsf{lv}(S)$ and $\gamma_{\ell_1} = \gamma''_{\ell_1} \cdot i_{S_2} \cdot f_S$, we also have that $|\gamma_{\ell_1}| \leq 2 \cdot \mathsf{lv}(S)$. Similarly, by the inductive hypothesis, for each $\ell_2 \in [j_1+1..n]$, we have that $|\gamma''_{\ell_2}| \leq 2 \cdot \mathsf{lv}(S_2)$. Since $\mathsf{lv}(S_2) < \mathsf{lv}(S)$ and $\gamma_{\ell_2} = \gamma''_{\ell_2} \cdot f_S$, we also have that $|\gamma_{\ell_2}| \leq 2 \cdot \mathsf{lv}(S)$. Since $\gamma_0 = \epsilon$, for each $i \in [0..n + 1]$, we have that $|\gamma_i| \leq 2 \cdot \mathsf{lv}(S)$.

(iii) By the inductive hypothesis, we have that $S_1 \Rightarrow^* \rho_1 - \rho_{j_1}$ and $S_2 \Rightarrow^* \rho_{j_1+1} - \rho_n$. Given that $S \Rightarrow^* S_1 \cdot S_2$, that $\rho_1 = \rho_0$, and that $\rho_{n+1} = \rho_n$, we conclude that $S \Rightarrow^* \rho_0 - \rho_{n+1}$.

There are no other possibilities for the form of (7), so the claim of this lemma holds for each derivation of that form. □

We are finally ready to show that PDA $\mathcal{P}_S$ satisfies properties (1) and (2).

Lemma 13. *For each role $S \in \Sigma_{\mathcal{R}}$ and each role chain ρ, we have that*

1. *$\rho \in \mathcal{L}(\mathcal{P}_S)$ implies $\rho \in \mathcal{L}(S)$, and*
2. *$\mathcal{P}_S$ has stack bounded by $2 \cdot \mathsf{lv}(S) + 1$.*

Proof. By the definition of $\mathcal{P}_S$, transitions resulting from case p in Definition 4 are the only ones popping elements from the stack, and these never pop symbol $\bot$; hence, at each point i in an accepting derivation of $\mathcal{P}_S$, the stack content γ_i is of the form $\gamma_i := \gamma'_i \cdot \bot$. Then, the two claims follow immediately from Lemma 12. □

3.1.2 Completeness

We next prove that our encoding is also complete, thus proving Theorem 8.

Lemma 14. *For each role $S \in \Sigma_{\mathcal{R}}$ and each role chain ρ, we have that $\rho \in \mathcal{L}(S)$ implies $\rho \in \mathcal{L}(\mathcal{P}_S)$.*

	Axiom Type	Derivation
(t1)	$\epsilon \sqsubseteq T$	$\langle i_T, \rho'', \gamma_i\rangle \vdash_{t1} \langle f_T, \rho'', \gamma_i\rangle$
(t2)	$T \cdot T \sqsubseteq T$	$\langle i_T, T \cdot T \cdot \rho'', \gamma_i\rangle \vdash_r \langle f_T, T \cdot \rho'', \gamma_i\rangle \vdash_{t2}$
		$\langle i_T, T \cdot \rho'', \gamma_i\rangle \vdash_r \langle f_T, \rho'', \gamma_i\rangle$
(t3)	$T_1 \cdot T \sqsubseteq T$	$\langle i_T, T_1 \cdot T \cdot \rho'', \gamma_i\rangle \vdash_{t3}$
		$\langle i_{T_1}, T_1 \cdot T \cdot \rho'', i_T \cdot \gamma_i\rangle \vdash_r \langle f_{T_1}, T \cdot \rho'', i_T \cdot \gamma_i\rangle \vdash_p$
		$\langle i_T, T \cdot \rho'', \gamma_i\rangle \vdash_r \langle f_T, \rho'', \gamma_i\rangle$
(t4)	$T_1 \cdot T_2 \sqsubseteq T$	$\langle i_T, T_1 \cdot T_2 \cdot \rho'', \gamma_i\rangle \vdash_{t4}$
		$\langle i_{T_1}, T_1 \cdot T_2 \cdot \rho'', i_{T_2} \cdot f_T \cdot \gamma_i\rangle \vdash_r \langle f_{T_1}, T_2 \cdot \rho'', i_{T_2} \cdot f_T \cdot \gamma_i\rangle \vdash_p$
		$\langle i_{T_2}, T_2 \cdot \rho'', f_T \cdot \gamma_i\rangle \vdash_r \langle f_{T_2}, \rho'', f_T \cdot \gamma_i\rangle \vdash_p$
		$\langle f_T, \rho'', \gamma_i\rangle$
(t5)	$T \cdot T_2 \sqsubseteq T$	$\langle i_T, T \cdot T_2 \cdot \rho'', \gamma_i\rangle \vdash_r \langle f_T, T_2 \cdot \rho'', \gamma_i\rangle \vdash_{t5}$
		$\langle i_{T_2}, T_2 \cdot \rho'', f_T \cdot \gamma_i\rangle \vdash_r \langle f_{T_2}, \rho'', f_T \cdot \gamma_i\rangle \vdash_p$
		$\langle f_T, \rho'', \gamma_i\rangle$

Table 4: Definition of derivation (9) depending on the form of axiom $\rho \sqsubseteq T$.

Proof. Consider an arbitrary role $S \in \Sigma_{\mathcal{R}}$. In the following, for each role chain ρ, we write $S \overset{0}{\Rightarrow} \rho$ if $\rho = S$; furthermore, for each $m \in \mathbb{N}^+$, we write $S \overset{m}{\Rightarrow} \rho$ if role chains $\rho_1, \ldots, \rho_m$ exist such that $S \Rightarrow \rho_1 \Rightarrow \cdots \Rightarrow \rho_m$ and $\rho_m = \rho$. By the definition of $\mathcal{L}(S)$, we have that $\rho \in \mathcal{L}(S)$ if and only if a natural number $m \in \mathbb{N}$ exists such that $S \overset{m}{\Rightarrow} \rho$. By induction on $m \in \mathbb{N}$, we next show that $S \overset{m}{\Rightarrow} \rho$ implies $\rho \in \mathcal{L}(\mathcal{P}_S)$.

Base case. Let $m = 0$. Then, we have that $S \overset{0}{\Rightarrow} S$. We consider two cases depending on the form of role $S \in \Sigma_{\mathcal{R}}$.

- $S = \top_r$. By case (ur) in Definition 4, we have that $\langle i_{\top_r}, \top_r, \bot\rangle \vdash_{ur} \langle f_{\top_r}, \epsilon, \bot\rangle$.
- $S \in \Sigma_{\mathcal{R}} \setminus \{\top_r\}$. By case (r) in Definition 4, we have $\langle i_S, S, \bot\rangle \vdash_r \langle f_S, \epsilon, \bot\rangle$.

In either case we have $S \in \mathcal{L}(\mathcal{P}_S)$, as required.

Inductive step. Consider an arbitrary $m \in \mathbb{N}$ and assume that, for each role chain ρ' such that $S \overset{m}{\Rightarrow} \rho'$, we have $\rho' \in \mathcal{L}(\mathcal{P}_S)$; we show that the same holds for $m+1$. Then, consider arbitrary role chains $\rho_1, \ldots, \rho_{m+1}$ such that $S \Rightarrow \rho_1 \Rightarrow \cdots \Rightarrow \rho_m \Rightarrow \rho_{m+1}$. By the definition of relation $\Rightarrow$, a role $T \in \Sigma_{\mathcal{R}}$ and role chains ρ', ρ, ρ'' exist such that role chain ρ_m is of the form $\rho_m := \rho' \cdot T \cdot \rho''$, role chain ρ_{m+1} is of the form $\rho_{m+1} := \rho' \cdot \rho \cdot \rho''$, and $T \Rightarrow \rho$. Since $S \overset{m}{\Rightarrow} \rho' \cdot T \cdot \rho''$, by the inductive hypothesis, we have $\rho' \cdot T \cdot \rho'' \in \mathcal{L}(\mathcal{P}_S)$, so a sequence $\langle s_0, \rho_0, \gamma_0\rangle \vdash \cdots \vdash \langle s_n, \rho_n, \gamma_n\rangle$ of $\mathcal{P}_S$ exists with $s_0 = i_S$ and $s_n = f_S$; furthermore, $\gamma_0 = \bot$ and $\gamma_n = \bot$; finally, $\rho_0 = \rho' \cdot T \cdot \rho''$ and $\rho_n = \epsilon$. Then there exists an index $i \in [0..n-1]$ such that $\rho_i = T \cdot \rho''$ and $\rho_{i+1} = \rho''$. Furthermore, for each $j \in [0..i]$, role chain ρ_j is of the form $\rho_j := \rho_j^- \cdot T \cdot \rho''$ for some role chain $\rho_j^- \in \mathsf{R}^*$. Next, consider the form of x_i in $\langle s_i, \rho_i, \gamma_i\rangle \vdash_{x_i} \langle s_{i+1}, \rho_{i+1}, \gamma_{i+1}\rangle$. By Definition 4, only transitions in cases (r) and (ur) read symbols from the input, so $x_i \in \{r, ur\}$. We show that the lemma holds in each case.

(*Case 1*) Consider the case in which $x_i = r$. Then, we have $s_i = i_T$ and $s_{i+1} = f_T$, $\gamma_i = \gamma_{i+1}$, and $T \in \Sigma_{\mathcal{R}} \setminus \{\top_r\}$. Due to $T \Rightarrow^* \rho$ and $T \neq \top_r$, we have $\rho \sqsubseteq T \in \mathcal{R}$. Then, the following is also a derivation of $\mathcal{P}_S$

$$\langle s_0,\ \rho_0^- \cdot \rho \cdot \rho'',\ \gamma_0\rangle \vdash \cdots \vdash \langle s_i,\ \rho_i^- \cdot \rho \cdot \rho'',\ \gamma_i\rangle \vdash^* \tag{8}$$

$$[\text{The derivation from Table 4 for } \rho \sqsubseteq T] \vdash^* \tag{9}$$

$$\langle s_{i+1},\ \rho'',\ \gamma_{i+1}\rangle \vdash \cdots \vdash \langle s_n,\ \epsilon,\ \gamma_n\rangle \tag{10}$$

where the derivation in (9) is defined in Table 4 depending on the form of axiom $\rho \sqsubseteq T \in \mathcal{R}$.

(*Case 2*) Consider the case in which $x_i = ur$. Then, we have $s_i = i_{\top_r}$ and $s_{i+1} = f_{\top_r}$, $\gamma_i = \gamma_{i+1}$ and $T \in \Sigma_{\mathcal{R}}$. Then, the following is also a derivation of $\mathcal{P}_S$

$$\langle s_0,\ \rho_0^- \cdot \rho \cdot \rho'',\ \gamma_0\rangle \vdash \cdots \vdash \langle s_i,\ \rho_i^- \cdot \rho \cdot \rho'',\ \gamma_i\rangle \vdash^* \tag{11}$$

$$[\text{The derivation } \mathsf{seq}(\rho, \rho'', \gamma_i) \text{ in (14)}] \vdash^* \tag{12}$$

$$\langle s_{i+1},\ \rho'',\ \gamma_{i+1}\rangle \vdash \cdots \vdash \langle s_n,\ \epsilon,\ \gamma_n\rangle \tag{13}$$

where the derivation $\mathsf{seq}(\rho, \rho', \gamma_i)$ in (12) is inductively defined as follows.

$$\mathsf{seq}(\rho, \rho'', \gamma_i) := \begin{cases} \langle i_{\top_r}, \rho'', \gamma_i\rangle \vdash_{u1} \langle f_{\top_r}, \rho'', \gamma_i\rangle & \text{if } \rho = \epsilon, \\ \langle i_{\top_r}, \rho \cdot \rho'', \gamma_i\rangle \vdash_{ur} \langle f_{\top_r}, \bar{\rho} \cdot \rho'', \gamma_i\rangle \vdash_{u2} \mathsf{seq}(\bar{\rho}, \rho'', \gamma_i) & \text{if } \rho = P \cdot \bar{\rho}. \end{cases} \tag{14}$$

Therefore, in either case, we have $\rho' \cdot \rho \cdot \rho'' \in \mathcal{L}(\mathcal{P}_S)$, as required. □

4. A Polynomial Space BCQ Answering Algorithm for $\mathcal{ELRO}^+$

Each $\mathcal{ELRO}^+$ knowledge base $\mathcal{K}$ can be translated into a set of first-order Horn clauses, so a Boolean CQ q over $\mathcal{K}$ can be answered by evaluating q over a so-called *canonical model*—a model that can be homomorphically embedded into any other model of $\mathcal{K}$. Canonical models are usually obtained using *chase*. Many different chase variants have been studied in the literature, each producing a different, but homomorphically equivalent, canonical model (Johnson & Klug, 1984; Marnette, 2009; Calì, Gottlob, & Kifer, 2013; Baget, Leclère, Mugnier, & Salvat, 2011). In this paper, we introduce a variant that we call *consequence-based chase*, and the (possibly infinite) set of assertions $\mathcal{I}_{\mathcal{K}}$ it produces on $\mathcal{K}$ we call the *universal interpretation* of $\mathcal{K}$. To compute $\mathcal{I}_{\mathcal{K}}$, consequence-based chase initialises $\mathcal{I}_{\mathcal{K}}$ to contain the ABox of $\mathcal{K}$, as well as assertions $\{a\}(a)$, $\top_c(a)$, and $\top_r(a, b)$ for all individuals a and b occurring in $\mathcal{K}$; then, it iteratively extends $\mathcal{I}_{\mathcal{K}}$ using chase rules. The slightly unusual aspect of our chase variant is that it considers the axioms entailed by (and not only contained in) $\mathcal{K}$. For example, if $\mathcal{I}_{\mathcal{K}}$ at some point contains assertion $A(w)$ and $\mathcal{K} \models A \sqsubseteq \exists S.B$ holds, then $\mathcal{I}_{\mathcal{K}}$ is extended with assertions $S(w, w')$ and $B(w')$ where w' is a fresh term; term w' is said to be *auxiliary* and have *type* S, B and *concept type* B. The BCQ answering algorithm we present in this section is based on checking consequences of $\mathcal{K}$, so our chase variant makes our proofs simpler. Example 15 illustrates these aspects.

Example 15. *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \emptyset\rangle$ be an $\mathcal{ELRO}^+$ KB, where $\mathcal{T}$ contains axioms* (15)–(21), *and $\mathcal{R}$ contains role inclusion* (22).

$$\{a\} \sqsubseteq \exists S.A \tag{15}$$

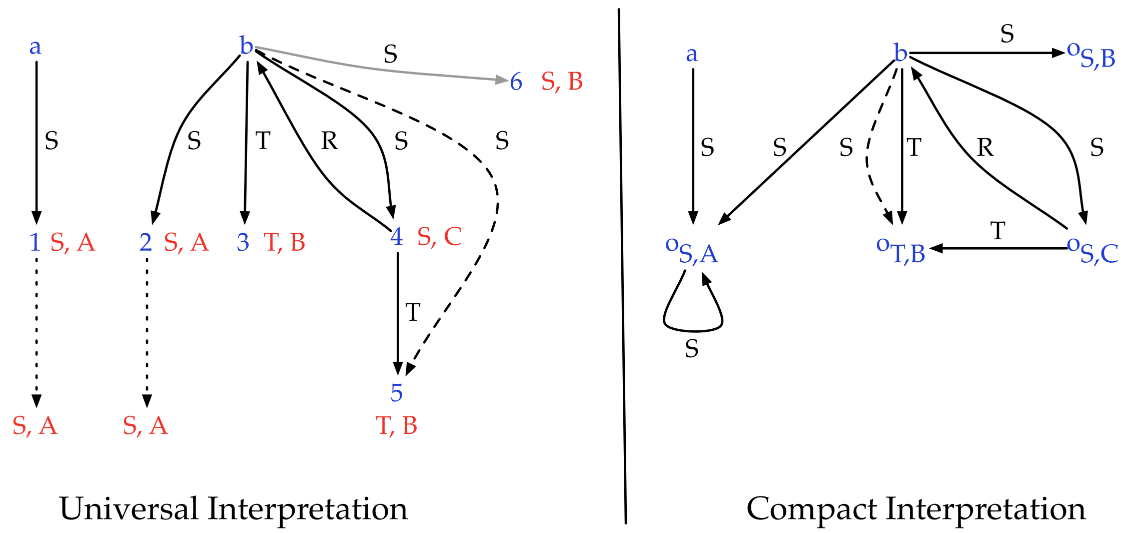

Figure 4: The universal interpretation and the compact interpretation for $\mathcal{K}$

$$A \sqsubseteq \exists S.A \tag{16}$$
$$\{b\} \sqsubseteq \exists S.A \tag{17}$$
$$\{b\} \sqsubseteq \exists T.B \tag{18}$$
$$\{b\} \sqsubseteq \exists S.C \tag{19}$$
$$C \sqsubseteq \exists T.B \tag{20}$$
$$C \sqsubseteq \exists R.\{b\} \tag{21}$$
$$S \cdot T \sqsubseteq S \tag{22}$$

Figure 4 shows the universal interpretation $\mathcal{I}_\mathcal{K}$ of $\mathcal{K}$. Assertions involving $\top_c$ and $\top_r$ are not shown for clarity. The edges obtained via role inclusions are dashed; remaining edges are solid, apart from the dotted edges, which denote repetition of solid edges. Black edges are obtained using conventional chase variants, whereas the light grey subbranches of $\mathcal{I}_\mathcal{K}$ are caused by axioms entailed by, but not occurring in, $\mathcal{K}$. Auxiliary terms are labelled using integers, and the term's type is shown next to each term. Universal interpretation $\mathcal{I}_\mathcal{K}$ can be viewed as a family of directed trees whose roots are the individuals in $\mathcal{K}$ and where solid edges point from parents to children or to the individuals in $\mathcal{K}$. Axiom (16) makes $\mathcal{I}_\mathcal{K}$ infinite, so a decision procedure for BCQ answering cannot simply materialise $\mathcal{I}_\mathcal{K}$ and then evaluate the query in it; instead, a finitary representation of $\mathcal{I}_\mathcal{K}$ is needed.

By axioms (19), (20), and (22), we have $\mathcal{K} \models \{b\} \sqsubseteq \exists S.B$; but then, since $\{b\}(b) \in \mathcal{I}_\mathcal{K}$, the consequence-based chase ensures that $\{S(b,6), B(6)\} \subseteq \mathcal{I}_\mathcal{K}$ holds as well. In contrast, commonly considered chase variants do not ensure $\{S(b,6), B(6)\} \subseteq \mathcal{I}_\mathcal{K}$ because $\mathcal{K}$ does not contain axiom $\{b\} \sqsubseteq \exists S.B$.

In the rest of this section, we present the first worst-case optimal algorithm that decides $\mathcal{K} \models q$ given an arbitrary regular $\mathcal{ELRO}^+$ KB $\mathcal{K}$ and a Boolean CQ q over $\mathcal{K}$. Towards this goal, in Section 4.1 we review the existing approaches to answering CQs in DLs and discuss why these techniques do not provide an optimal procedure for $\mathcal{ELRO}^+$; in Section 4.2 we discuss the intuitions behind our algorithm; in Section 4.3 we introduce the algorithm formally and show that it runs in polynomial space in the combined size of $\mathcal{K}$ and q and in polynomial time in the size of $\mathcal{K}$; and in Section 4.4 we prove the algorithm's correctness.

4.1 Existing Approaches to Answering CQs

Techniques for answering conjunctive queries over DL knowledge bases developed thus far can be broadly classified into the following three groups.

The first group consists of automata-based approaches for DLs such as Horn-$\mathcal{SHIQ}$ and Horn-$\mathcal{SROIQ}$ (Ortiz et al., 2011), $\mathcal{SH}$ (Eiter, Ortiz, & Simkus, 2012a), and the fragment of $\mathcal{ELRO}^+$ obtained by disallowing the universal role, reflexive roles, and self restrictions (Krötzsch et al., 2007). All these techniques, however, require constructing automata whose size can be exponential in the size of the knowledge base.

The second group consists of rewriting-based approaches. Roughly speaking, these approaches rewrite the query and/or the TBox into another formalism, usually a union of CQs or a datalog program; the relevant answers can then be obtained by evaluating the rewriting over the ABox. Rewriting-based approaches have been proposed for members of the DL-Lite family (Artale et al., 2009; Calvanese et al., 2007), the DLs $\mathcal{EL}$ (Rosati, 2007), $\mathcal{ELHIO}_\bot$ (Pérez-Urbina et al., 2010; Mora, Rosati, & Corcho, 2014) and Horn-$\mathcal{SHIQ}$ (Eiter, Ortiz, Simkus, Tran, & Xiao, 2012b), and members of the datalog$^\pm$ family (Virgilio, Orsi, Tanca, & Torlone, 2012), to name just a few. No rewriting approach, however, supports for both nominals and role inclusions. Moreover, a common shortcoming is that rewritings can be exponential in the query and/or TBox size, so these approaches may also use exponential space.

The third group consists of approaches based on a particular interpretation of $\mathcal{K}$ that we call the *compact interpretation*. Figure 4 shows this interpretation for the KB $\mathcal{K}$ from Example 15: it finitely approximates the universal interpretation by using individuals of the form $o_{S,B}$ to represent all auxiliary terms of type S, B. The compact interpretation can thus be materialised in space polynomial in $|\mathcal{K}|$, and it can be used to answer instance queries and test atomic subsumptions over $\mathcal{K}$ (Baader et al., 2005; Krötzsch, 2011). Materialising the compact interpretation lies at the core of many reasoning algorithms for $\mathcal{EL}$ variants, so it is natural to try to use this interpretation for answering CQs as well. Since the compact interpretation is a model of $\mathcal{K}$, each CQ that maps on the universal interpretation maps on the compact interpretation as well; however, as Example 16 shows, the converse does not necessarily hold.

Example 16. *Let $\mathcal{K}$ be as in Example 15, and let q_1, q_2, and q_3 be the following BCQs.*

$$q_1 = \exists x.\ R(x, b) \qquad q_2 = \exists x.\ S(a, x) \wedge S(b, x) \qquad q_3 = \exists x.\ T(b, x) \wedge S(b, x)$$

The compact interpretation for $\mathcal{K}$ is shown in Figure 4; as one can see, it is obtained from the universal interpretation by merging all terms of type S, B onto the individual $o_{S,B}$. Now query q_1 can be mapped onto both the compact and the universal interpretation, while queries q_2 and q_3 can be mapped only onto the compact interpretation. Thus, evaluating q_2 and q_3 over the compact interpretation produces unsound answers.

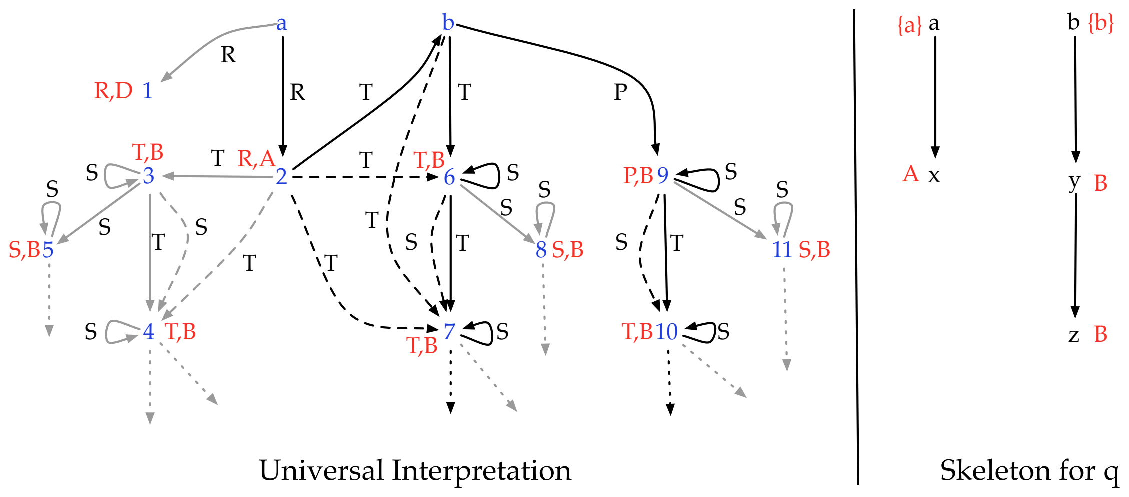

Figure 5: The universal interpretation of $\mathcal{K}$ and the skeleton for q

As a remedy, *combined approaches* were developed that first evaluate the query in the compact interpretation and then filter the results to eliminate unsound answers. Such approaches have been developed for members of the DL-Lite family (Kontchakov et al., 2011; Lutz, Seylan, Toman, & Wolter, 2013) and the $\mathcal{EL}$ family (Lutz, Toman, & Wolter, 2009; Stefanoni et al., 2013) of DLs, and the datalog$^{\pm}$ family (Gottlob, Manna, & Pieris, 2014) of rule-based languages. In particular, Stefanoni et al. (2013) developed a filtering step applicable to the DL $\mathcal{ELHO}_{\bot}^{dr}$, but this step breaks down if $\mathcal{K}$ contains role inclusions. Query q_3 from Example 16 can be mapped onto the compact interpretation by mapping atom $S(b,x)$ to a dashed edge (i.e., to an edge obtained via role inclusions); moreover, q_3 is tree-shaped, and so the filtering step by Stefanoni et al. (2013) does not identify this match as unsound. This problem can be intuitively understood as follows. By 'unfolding' the query by (22), query q_3 essentially asks whether role chains $\rho_1 \in \mathcal{L}(S)$ and $\rho_2 \in \mathcal{L}(T)$ exist that label a path of solid edges in $\mathcal{I}_{\mathcal{K}}$ starting at b. In the compact interpretation, this is satisfied by $\rho_1 = S \cdot T$ and $\rho_2 = T$ when x is mapped to individual $o_{T,B}$. Individual $o_{T,B}$, however, represents distinct terms 3 and 5 from $\mathcal{I}_{\mathcal{K}}$; hence, although 3 is connected to b via ρ_1 and 5 is connected to b via ρ_2, role chains ρ_1 and ρ_2 do not satisfy query q_3. In other words, the compact interpretation is 'too small' to represent the relevant conditions.

4.2 Intuitions

Our worst-case optimal procedure for BCQ answering in $\mathcal{ELRO}^+$ is shown in Algorithm 1 on page 681. It essentially extends and refines the algorithm by Krötzsch et al. (2007). We explain the underlying intuitions using the knowledge base shown in Example 17.

Example 17. *Let $\mathcal{K}$ be the $\mathcal{ELRO}^+$ knowledge base whose TBox contains axioms* (23)–(29) *and whose RBox contains role inclusions* (30)–(31).

$$\{a\} \sqsubseteq \exists R.A \tag{23}$$

$$A \sqsubseteq \exists T.\{b\} \tag{24}$$

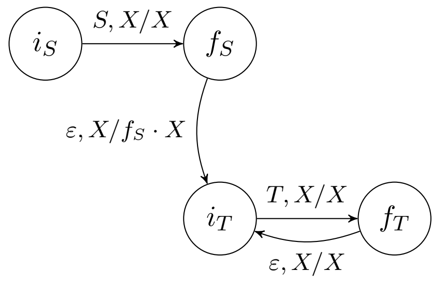

Figure 6: The transitions of $\delta_{\mathcal{R}}$ corresponding to axioms (30)–(31)

$$A \sqsubseteq D \tag{25}$$
$$\{b\} \sqsubseteq \exists T.B \tag{26}$$
$$\{b\} \sqsubseteq \exists P.B \tag{27}$$
$$B \sqsubseteq \exists S.\mathsf{Self} \tag{28}$$
$$B \sqsubseteq \exists T.B \tag{29}$$
$$T \cdot T \sqsubseteq T \tag{30}$$
$$S \cdot T \sqsubseteq S \tag{31}$$

Moreover, let q be the following Boolean CQ over $\mathcal{K}$.

$$q = \exists x, y, z.\ D(x) \wedge T(x, z) \wedge S(y, z) \tag{32}$$

Figure 5 shows the universal interpretation $\mathcal{I}_{\mathcal{K}}$ for $\mathcal{K}$; notation is as in Example 15. The solid looping edges on auxiliary terms of concept type B are obtained from axiom (28). One can see that $\mathcal{K} \models q$ holds; for example, the following substitution τ embeds q into $\mathcal{I}_{\mathcal{K}}$.

$$\tau = \{x \mapsto 2,\ y \mapsto 6,\ z \mapsto 7\} \tag{33}$$

Our algorithm uses the PDA encoding of the RBox described in Section 3. The transition function $\delta_{\mathcal{R}}$ for axioms (30)–(31) is shown in Figure 6; notation is the same as in Example 6; and note that Figure 6 is contained in Figure 3.

We must prove the existence of a substitution τ mapping q into $\mathcal{I}_{\mathcal{K}}$. Such a substitution τ can map the binary atoms in q to the dashed edges in Figure 5. Dashed edges introduce shortcuts between terms in $\mathcal{I}_{\mathcal{K}}$, but each dashed edge can be unfolded into a path consisting only of solid edges using the role inclusions in $\mathcal{K}$. The solid paths in $\mathcal{I}_{\mathcal{K}}$ can be of two types: auxiliary paths involve only auxiliary terms, whereas nominal paths require moving through at least one individual. For instance, edge $T(2, 7)$ can be unfolded into the path $\rho_T = T \cdot T \cdot T$ connecting 2 with b, b with 6, and 6 with 7. In contrast, edge $S(6, 7)$ can be unfolded into the path $\rho_S = S \cdot T$ connecting 6 with itself, and 6 with 7. Our algorithm then uses PDAs with transition function from Figure 6 to represent each binary atom in q as a sequence of binary atoms to be mapped over the corresponding solid path in $\mathcal{I}_{\mathcal{K}}$. Interpretation $\mathcal{I}_{\mathcal{K}}$, however, is infinite, so the space of possible substitutions is also infinite. Hence, to prove the existence of a substitution τ mapping q into $\mathcal{I}_{\mathcal{K}}$, we cannot simply enumerate all of them, and we use Algorithm 1 instead.

In line 1 we check whether $\mathcal{K}$ is unsatisfiable; if so, then $\mathcal{K} \models q$ holds trivially. Next, in line 2 we guess a substitution σ and continue checking $\mathcal{K} \models \sigma(q)$; thus, this step takes into account that variables could be mapped to individuals, and that two variables could be mapped to the same term. In our example, we can guess σ to be the identity mapping on $\vec{y}$. In step 3, we then guess a finite structure, called the *skeleton for* $\sigma(q)$, which represents a (possibly infinite) set of substitutions mapping the variables of $\sigma(q)$ to distinct auxiliary terms in $\mathcal{I}_{\mathcal{K}}$. Figure 5 shows the skeleton $\mathcal{S}$ for the query in Example 17: skeleton vertices are the individuals of $\mathcal{K}$ and the variables of $\sigma(q)$, and they are arranged as a forest whose roots are the individuals; moreover, each vertex v of $\mathcal{S}$ is assigned an atomic concept $\kappa(v)$. After this step, skeleton $\mathcal{S}$ represents each substitution τ (if any) satisfying the following two properties:

1. τ maps each variable x to a term of concept type $\kappa(x)$, and
2. for each edge $\langle v, v' \rangle$ in $\mathcal{S}$, we have that $\tau(v')$ is a descendant of $\tau(v)$ in $\mathcal{I}_{\mathcal{K}}$.

We next extend $\mathcal{S}$ with conditions that prune this set of substitutions, with the goal of leaving only substitutions *compatible* with $\sigma(q)$—that is, that embed $\sigma(q)$ into $\mathcal{I}_{\mathcal{K}}$.

We establish compatibility with the unary atoms of $\sigma(q)$ in line 4. In particular, consider atom $D(x)$ in $\sigma(q)$. By property (1), each substitution τ represented by the skeleton in Figure 5 maps variable x to a term of concept type $\kappa(x) = A$, implying that $A(\tau(x)) \in \mathcal{I}_{\mathcal{K}}$ holds. But then, since $\mathcal{K} \models \kappa(x) \sqsubseteq D$ holds, we know that $D(\tau(x)) \in \mathcal{I}_{\mathcal{K}}$ holds as well; thus, atom $D(x)$ is satisfied for each substitution represented by $\mathcal{S}$.

In contrast, we cannot establish compatibility of binary query atoms using entailment checking only, because vertex labels and the relative position of vertices do not sufficiently describe the substitutions. For example, substitution

$$\tau_1 = \{x \mapsto 2,\, y \mapsto 9,\, z \mapsto 10\} \tag{34}$$

satisfies properties (1) and (2), but $T(\tau_1(x), \tau_1(z)) \notin \mathcal{I}_{\mathcal{K}}$.

To prune such substitutions, in lines 5–16 of Algorithm 1 we guess for each binary atom in $\sigma(q)$ how to unfold it as a sequence of solid steps in $\mathcal{I}_{\mathcal{K}}$. As solid paths in $\mathcal{I}_{\mathcal{K}}$ can go through nominals or through auxiliary terms only, the two possibilities are accounted for by the guessing in line 8. Moreover, the skeleton already constrains the relative positions of query terms, so we represent the unfolding of each binary atom by labelling each edge $\langle v, v' \rangle$ in $\mathcal{S}$ with a set $L(v, v')$ of bounded-stack PDAs with transition function from Figure 6 so that each substitution τ represented by $\mathcal{S}$ also satisfies the following property:

3. for each PDA $\mathcal{P} \in L(v, v')$, a nonempty role chain $\rho \in \mathcal{L}(\mathcal{P})$ exists labelling a path in $\mathcal{I}_{\mathcal{K}}$ over solid edges from $\tau(v)$ to $\tau(v')$.

We next illustrate how to label the edges of $\mathcal{S}$ so that all substitutions satisfying properties (1)–(3) are compatible with each binary atom of $\sigma(q)$.

For $T(x, z)$, we must ensure that, for each substitution τ represented by $\mathcal{S}$, a role chain $\rho_T \in \mathcal{L}(T)$ exists connecting $\tau(x)$ to $\tau(z)$ using only solid edges. Since the relative positions of $\tau(x)$ and $\tau(z)$ in $\mathcal{I}_{\mathcal{K}}$ is determined by $\mathcal{S}$ as shown in Figure 5, such a path must connect $\tau(x)$ with b, then connect b with $\tau(y)$, and finally connect $\tau(y)$ with $\tau(z)$. In addition, we can assume that no individuals occur on paths from b to $\tau(y)$, and from $\tau(y)$ to $\tau(z)$: if a

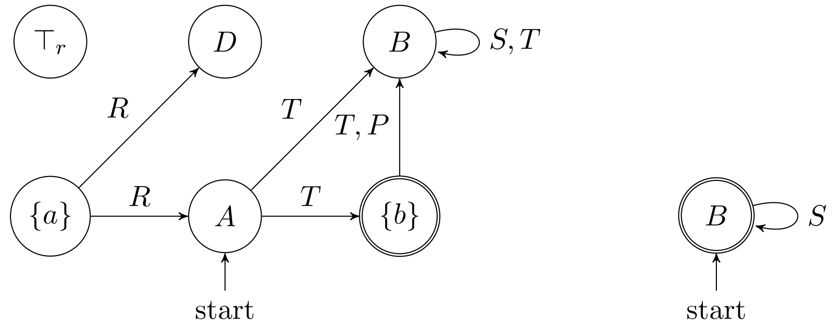

Figure 7: On the left-hand side, the walking finite automaton $\mathsf{wfa}(A,\ \{b\})$ (transitions involving $\top_c$ and $\top_r$ are not shown for clarity); on the right-hand side, the stationary finite automaton $\mathsf{sfa}(B)$

path from $\tau(x)$ to $\tau(z)$ involves individuals other than b or if it visits b more than once, we can 'absorb' all such path segments into the subpath from $\tau(x)$ to b. Thus, we check the existence of such ρ_T by setting $v_0 = a_u = b$ in lines 7–8 (no guessing is possible in line 8 in this case) and 'splitting' in lines 7–11 the path ρ_T into the three subpaths. In particular, in line 10 we guess states s_0, s_1, and $s_2 = f_T$ and in line 11 we guess stack words γ_0, γ_1, and $\gamma_2 = \bot$ with the following properties:

(i) the subpaths from $\tau(x)$ to b are described by PDA $\mathcal{P}_T^0$ whose start state and stack are i_T and $\bot$, respectively, and the final state and stack are s_0 and γ_0, respectively;

(ii) the subpaths from b to $\tau(y)$ are described by PDA $\mathcal{P}_T^1$ whose start state and stack are s_0 and γ_0, respectively, and the final state and stack are s_1 and γ_1, respectively; and

(iii) the subpaths from $\tau(y)$ to $\tau(z)$ are described by PDA $\mathcal{P}_T^2$ whose start state and stack are s_1 and γ_1, respectively, and the final state and stack are s_2 and γ_2, respectively.

We do not know what terms of $\mathcal{I}_\mathcal{K}$ variables y and z should be mapped to, so we cannot check the existence of paths in (ii) and (iii) independently. Therefore, we add in line 12 PDAs $\mathcal{P}_T^1$ and $\mathcal{P}_T^2$ as constraints on edges $\langle b, y\rangle$ and $\langle y, z\rangle$ in $\mathcal{S}$, respectively. The edges of $\mathcal{S}$ thus 'accumulate' all constraints that the moves to auxiliary terms must satisfy; later we shall explain how in lines 17–18 we check these constraints and, if this check passes, how we know that we can map y and z to auxiliary terms whose concept types are $\kappa(y)$ and $\kappa(z)$, respectively. In contrast, the path in (i) finishes in an individual, so we can check existence of such a path independently from any other constraint. To this end, we construct the *walking finite automaton* $\mathsf{wfa}(A,\ \{b\})$ shown on the left-hand side of Figure 7. Such $\mathsf{wfa}(A,\ \{b\})$ describes the moves in $\mathcal{I}_\mathcal{K}$ from terms with concept type $\kappa(x) = A$ to individual b—that is, such that $\rho \in \mathcal{L}(\mathsf{wfa}(A,\ \{b\}))$ for each term w with concept type A and for each role chain ρ connecting w to b in $\mathcal{I}_\mathcal{K}$ via solid edges; then, in line 14 we check whether the intersection of the languages of $\mathsf{wfa}(A,\ \{b\})$ and $\mathcal{P}_T^0$ is empty. As $\mathsf{wfa}(A,\ \{b\})$ is a FA and $\mathcal{P}_T^0$ is a PDA, we can test the emptiness of the intersection of their languages in polynomial time (Hopcroft et al., 2003). In our example, we can guess $s_0 = s_1 = s_2 = f_T$.

Thus, $\mathcal{P}_T^1$ accepts the language T^*, and because b is connected to $\tau_1(y)$ by a solid edge labelled by P, adding $\mathcal{P}_T^1$ as constraint on edge $\langle b, y\rangle$ ensures that substitution τ_1 from (34) does not satisfy property (3).

For $S(y, z)$, we must ensure that, for each substitution τ represented by $\mathcal{S}$, a role chain $\rho_S \in \mathcal{L}(S)$ exists connecting $\tau(y)$ to $\tau(z)$ using only solid edges. Now even though z is a descendant of y in $\mathcal{S}$, in line 8 we could guess $v_0 = a_u = b$, so that ρ_S connects $\tau(y)$ with b and then, without going through individuals, connects b with $\tau(z)$ via $\tau(y)$. In the rest of this paragraph, however, we consider the case in which ρ_S connects $\tau(y)$ with $\tau(z)$ directly, since that is the only possibility in our example, as one can see in Figure 5. Therefore, in line 8, we guess $v_0 = t = y$. But then, a path from $\tau(y)$ to $\tau(z)$ could first loop on $\tau(y)$ due to self-restrictions; then it must actually move from $\tau(y)$ to $\tau(z)$; and finally it could loop on $\tau(z)$. For reasons we discuss in the following paragraph, we 'absorb' the latter loop into a constraint added to edge $\langle y, z\rangle$; however, we check the existence of the former loop independently. Therefore, in lines 7–11 we 'split' ρ_S into two subpaths. In particular, in line 10 we guess states s_0 and $s_1 = f_S$, and in line 11 we guess stack words γ_0 and $\gamma_1 = \bot$ with the following properties:

(i) the looping on $\tau(y)$ is described by PDA $\mathcal{P}_S^0$ whose start state and stack are i_S and $\bot$, respectively, and the final state and stack are s_0 and γ_0, respectively; and

(ii) the subpaths from $\tau(y)$ to $\tau(z)$ that start with a move from $\tau(y)$, but possibly involve looping on $\tau(z)$, are described by PDA $\mathcal{P}_S^1$ whose start state and stack are s_0 and γ_0, respectively, and the final state and stack are s_1 and γ_1, respectively.

As in the previous case, we check (ii) by adding $\mathcal{P}_S^1$ as constraint on edge $\langle y, z\rangle$ in $\mathcal{S}$. Furthermore, we check the existence of a path in (i) by constructing the *stationary finite automaton* $\mathsf{sfa}(B)$ shown on the right-hand side of Figure 7. Such $\mathsf{sfa}(B)$ describes the possible loops on terms with concept type $\kappa(y) = B$; that is, such that $\rho \in \mathcal{L}(\mathsf{sfa}(B))$ for each term w with concept type B and for each role chain ρ corresponding to a (possibly empty) loop on w; then, in line 16 we check whether the intersection of the languages of $\mathsf{sfa}(B)$ and $\mathcal{P}_S^0$ is empty. In our example, we guess $s_0 = s_1 = f_S$; thus $\mathcal{P}_S^0$ accepts the language $S \cdot T^*$, whereas $\mathcal{P}_S^1$ accepts the language T^*.

Before line 17, skeleton $\mathcal{S}$ represents all substitutions that are compatible with the atoms in $\sigma(q)$, but we must still show that at least one such substitution can be realised by the universal interpretation $\mathcal{I}_\mathcal{K}$. To this end, we apply Algorithm 2 on page 681 to each edge $\langle v, v'\rangle$ in the skeleton, and thus we check whether terms $\tau(v)$ and $\tau(v')$ in $\mathcal{I}_\mathcal{K}$ exist that satisfy properties (1)–(3) for all PDAs in $L(v, v')$. Roughly speaking, we solve this problem by running all PDAs in parallel in lines 7–15 of Algorithm 2. However, we cannot materialise $\mathcal{I}_\mathcal{K}$, so we exploit a property of our consequence-based chase procedure: a term w with concept type A is connected to a term w' with concept type B in $\mathcal{I}_\mathcal{K}$ using a solid edge labelled by S if and only if $\mathcal{K} \models A \sqsubseteq \exists S.B$. Furthermore, the concept type of w fully describes the solid paths to descendants of w, so we do not need to keep track of the actual position in $\mathcal{I}_\mathcal{K}$; instead, we use variable $\mathsf{concept}$ to keep track of the current term's concept type. Thus, in line 9 we check existence of edges in $\mathcal{I}_\mathcal{K}$ via entailment checking; after that, for each PDA, in line 11 we guess a state s and a stack γ of the PDA, in line 12 we check whether the PDA can perform the move, and in line 14 we actually move the PDA. Due to self-restrictions and reflexive roles, however, the PDAs need not move in synchrony: after each move over a solid edge, each of the PDAs can independently loop on the current term. To this end, in line 11 we guess a state s' and a stack γ' that the PDA moves into after looping, and in line 13 we check whether the PDA can move from state s with stack γ to state s' with stack γ' using a role chain compatible with the concept type of the term the PDA is moving into, as given by the stationary finite automaton $\mathsf{sfa}(D)$. Because not all PDAs are required to loop, FA $\mathsf{sfa}(D)$ accepts the empty word. Algorithm 2 thus checks loops only *after* each move, which is why line 16 in Algorithm 1 is necessary. Lines 2–5 take into account that each of the PDAs is nondeterministic and so it can initially make several ε-transitions; note that an explicit check for ε-transitions is required only initially since line 13 allows for possible ε-transitions after each move along a solid edge. Finally, we ensure termination of Algorithm 2 by observing that, since the stack of each PDA in $L(v, v')$ is bounded, the number of current configurations of each of the PDA is exponential, and so the number of distinct tuples of the current PDA configurations is exponential as well; hence, the algorithm repeats computations after at most exponentially many steps. We thus obtain a nondeterministic decision procedure running in polynomial space by using a binary counter to stop the computation after all distinct configurations have been explored. For the constraints added to $\mathcal{S}$ in the previous paragraphs, one can check that Algorithm 2 returns true on all edges of $\mathcal{S}$; hence, $\mathcal{K} \models \sigma(q)$ holds, and thus $\mathcal{K} \models q$ holds as well.

4.3 Formalisation

We now formalise the intuitions from the previous section. Towards this goal, we fix a normalised $\mathcal{ELRO}^+$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ with a regular RBox $\mathcal{R}$, we let $Q_\mathcal{R}$, $\Gamma_\mathcal{R}$, and $\delta_\mathcal{R}$ be as specified in Definition 4, and we let $\mathsf{d}_\mathcal{R}$ be the depth of $\mathcal{R}$ as specified in Section 3. We start by formalising the notion of a skeleton of a Boolean CQ.

Definition 18. *A* skeleton for *a Boolean CQ $q = \exists \vec{y}.\ \psi(\vec{y})$ is a triple $\mathcal{S} = \langle V,\, E,\, \kappa \rangle$ with the following components.*

1. *$V = \mathsf{I}_\mathcal{K} \cup \vec{y}$ is the set of vertices.*
2. *$E \subseteq V \times \vec{y}$ is the set of edges such that the directed graph $\langle V, E \rangle$ is a forest whose roots are precisely the elements of $\mathsf{I}_\mathcal{K}$.*
3. *$\kappa : \vec{y} \mapsto \{\top_c\} \cup \mathsf{C}_\mathcal{K}$ is a function that maps the existential variables of q to atomic concepts. For convenience, κ is extended to V by $\kappa(v) := \{v\}$ for each $v \in \mathsf{I}_\mathcal{K}$.*

A path *in $\mathcal{S}$ is a nonempty sequence of (distinct) vertices $v_0, \ldots, v_n$ such that $n \geq 0$ and, for each $i \in [0..n-1]$, we have that $\langle v_i, v_{i+1} \rangle \in E$.*

Please observe that, as $\mathcal{K}$ is in normal form, there exists at least one individual occurring in $\mathcal{K}$ and thus $V \cap \mathsf{I} \neq \emptyset$. We next generalise the notion of a PDA encoding the RBox $\mathcal{R}$ from Definition 4 by allowing arbitrary start and final states as well as arbitrary start and final stacks of size at most $\mathsf{d}_\mathcal{R}$. These generalised PDA will be used in our algorithm to implement the splitting operation mentioned in Section 4.2.

Definition 19. *For states $s, s' \in Q_\mathcal{R}$ and words $\gamma, \gamma' \in \Gamma_\mathcal{R}^*$ with $|\gamma| \leq \mathsf{d}_\mathcal{R}$ and $|\gamma'| \leq \mathsf{d}_\mathcal{R}$, a* generalised PDA for $\mathcal{R}$ *is given by $\mathsf{pda}(s,\, \gamma,\, s',\, \gamma') := \langle Q_\mathcal{R},\, \Sigma_\mathcal{R},\, \Gamma_\mathcal{R},\, \delta_\mathcal{R},\, s,\, \gamma,\, s',\, \gamma' \rangle$.*

The following definition introduces automata that one can use to succinctly represent the axioms that logically follow from $\mathcal{K}$.

Definition 20. *Let A and B be basic concepts. The* walking finite automaton from A to B *is given by* $\mathsf{wfa}(A,\, B) := \langle Q,\, \Sigma_{\mathcal{R}},\, \delta_w,\, A,\, B\rangle$ *where Q and δ_w are as follows.*

- $Q = \{\top_c\} \cup \mathsf{C}_{\mathcal{K}} \cup \mathsf{N}_{\mathcal{K}}$.
- *δ_w is the transition function containing $D \in \delta_w(C, S)$ for each role $S \in \Sigma_{\mathcal{R}}$ and all states C and D in Q such that $\mathcal{K} \models C \sqsubseteq \exists S.D$.*

The stationary finite automaton for A *is given by* $\mathsf{sfa}(A) := \langle\{A\},\, \Sigma_{\mathcal{R}},\, \delta_s,\, A,\, A\rangle$ *where δ_s contains $A \in \delta_s(A, S)$ for each role $S \in \Sigma_{\mathcal{R}}$ with $\mathcal{K} \models A \sqsubseteq \exists S.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq S$.*

Boolean CQs can be answered by the nondeterministic procedure $\mathsf{entails}$ shown in Algorithm 1, which uses an auxiliary procedure exist shown in Algorithm 2. The following theorem states that $\mathsf{entails}(\mathcal{K}, q)$ decides $\mathcal{K} \models q$, and its proof is given in Section 4.4.

Theorem 21. *Let q be a Boolean CQ over $\mathcal{K}$. Then, $\mathcal{K} \models q$ if and only if a nondeterministic computation exists such that $\mathsf{entails}(\mathcal{K}, q)$ returns true.*

Finally, we determine the complexity of the algorithm $\mathsf{entails}$, and towards this goal we first determine the complexity of the auxiliary function exist.

Lemma 22. *Function $\mathsf{exist}(A,\, B,\, \{\mathcal{P}_j = \mathsf{pda}(s_j,\, \gamma_j,\, s'_j,\, \gamma'_j) \mid 0 \leq j \leq m\})$ can be implemented so that it uses space polynomial in $m \cdot |\mathcal{K}|$ and, if the RBox $\mathcal{R}$ is fixed, it runs in time polynomial in $|\mathcal{T}| + |\mathcal{A}|$.*

Proof. Consider arbitrary A, B, and $\mathcal{P}_j = \mathsf{pda}(s_j,\, \gamma_j,\, s'_j,\, \gamma'_j)$ as stated above; let M be as in Algorithm 2; and let $\vdash$ be the derivation relation corresponding to $\delta_{\mathcal{R}}$. By the definition of generalised PDAs, we have $|\gamma_j| \leq \mathsf{d}_{\mathcal{R}}$ and $|\gamma'_j| \leq \mathsf{d}_{\mathcal{R}}$ for each $j \in [1..m]$.

By Proposition 2, using polynomial time one can compute PDAs accepting languages $\mathcal{L}_{\mathsf{d}_{\mathcal{R}}}(\mathsf{pda}(s_j, \gamma_j, s, \gamma))$ and $\mathcal{L}_{\mathsf{d}_{\mathcal{R}}}(\mathsf{pda}(s, \gamma, s', \gamma'))$ in lines 4 and 13; therefore, checks in lines 4 and 13 can be implemented so that they use time (and therefore space) polynomial in $|\mathcal{K}|$ (Hopcroft et al., 2003, ch. 7).

For the space usage of Algorithm 2, please observe that the function stores the following information at each computation step:

(a) an array state of length m such that $\mathsf{state}[j] \in Q_{\mathcal{R}}$ for each $j \in [1..m]$, an array stack of length m such that $\mathsf{stack}[j] \in \Gamma^*_{\mathcal{R}}$ and $|\mathsf{stack}[j]| \leq \mathsf{d}_{\mathcal{R}}$ for each $j \in [1..m]$,

(b) a generalised PDA in line 4,

(c) a generalised PDA and a stationary automaton in line 14,

(d) a concept $\mathsf{concept} \in \{A, \top_c\} \cup \mathsf{C}_{\mathcal{K}}$ in line 1,

(e) a binary counter k such that $1 \leq k \leq M$, and

(f) the depth $\mathsf{d}_{\mathcal{R}}$ of $\mathcal{R}$, an atomic concept $D \in \{\top_c\} \cup \mathsf{C}_{\mathcal{K}}$, and a role $S \in \Sigma_{\mathcal{R}}$.

Algorithm 1: $\mathsf{entails}(\mathcal{K}, q)$

```
if $\mathcal{K}$ *is inconsistent* then return true
guess a substitution $\sigma$ with $\mathsf{dom}(\sigma) = \vec{y}$ and $\mathsf{rng}(\sigma) \subseteq \vec{y} \cup \mathsf{I}_{\mathcal{K}}$
guess a skeleton $\mathcal{S} = \langle V, E, \kappa \rangle$ for $\sigma(q)$
if *an atom* $A(t)$ *in* $\sigma(q)$ *exists such that* $\mathcal{K} \not\models \kappa(t) \sqsubseteq A$ then return false
foreach $\langle v, v' \rangle \in E$ do let $L(v, v') := \emptyset$
foreach *binary atom* $S(t, u)$ *in* $\sigma(q)$ do
    let $a_u$ be the unique individual such that $u$ is reachable from $a_u$ in $\mathcal{S}$
    guess $v_0 \in \{t, a_u\}$ such that $u$ is reachable from $v_0$ in $\mathcal{S}$
    let $v_0, \ldots, v_n$ be the unique path in $\mathcal{S}$ such that $v_n = u$
    guess states $s_0, \ldots, s_n$ in $Q_{\mathcal{R}}$ such that $s_n = f_S$
    guess words $\gamma_0, \ldots. \gamma_n$ in $\Gamma^*_{\mathcal{R}}$ such that $\gamma_n = \bot$ and $|\gamma_i| \leq \mathsf{d}_{\mathcal{R}}$ for each $i \in [0..n]$
    foreach $i \in [1..n]$ do let $L(v_{i-1}, v_i) := L(v_{i-1}, v_i) \cup \{\mathsf{pda}(s_{i-1}, \gamma_{i-1}, s_i, \gamma_i)\}$
    if $v_0 \in \mathsf{I}$ then
        if $\mathcal{L}(\mathsf{wfa}(\kappa(t), \kappa(v_0))) \cap \mathcal{L}(\mathsf{pda}(i_S, \bot, s_0, \gamma_0)) = \emptyset$ then return false
    else
        if $\mathcal{L}(\mathsf{sfa}(\kappa(v_0))) \cap \mathcal{L}(\mathsf{pda}(i_S, \bot, s_0, \gamma_0)) = \emptyset$ then return false
foreach $\langle v, v' \rangle \in E$ do
    if *not* $\mathsf{exist}(\kappa(v), \kappa(v'), L(v, v'))$ then return false
return true
```

Algorithm 2: $\mathsf{exist}(A, B, \{\mathcal{P}_j = \mathsf{pda}(s_j, \gamma_j, s'_j, \gamma'_j) \mid 0 \leq j \leq m\})$

```
let concept := $A$ and let $M := (1 + |\mathsf{C}_{\mathcal{K}}|) \cdot |Q_{\mathcal{R}}|^m \cdot (|\Gamma_{\mathcal{R}}|^{1+\mathsf{d}_{\mathcal{R}}})^m$
for $j = 1$ to $m$ do
    guess a state $s \in Q_{\mathcal{R}}$ and a word $\gamma \in \Gamma^*_{\mathcal{R}}$ such that $|\gamma| \leq \mathsf{d}_{\mathcal{R}}$
    if $\epsilon \notin \mathcal{L}_{\mathsf{d}_{\mathcal{R}}}(\mathsf{pda}(s_j, \gamma_j, s, \gamma))$ then return false
    set state$[j] := s$ and stack$[j] := \gamma$
guess $k \in \mathbb{N}$ such that $1 \leq k \leq M$
for $r = 1$ to $k$ do
    guess $S \in \Sigma_{\mathcal{R}}$ and $D \in \{\top_c\} \cup \mathsf{C}_{\mathcal{K}}$
    if $\mathcal{K} \not\models$ concept $\sqsubseteq \exists S.D$, *or* $\mathcal{K} \models D \sqsubseteq \{a\}$ *for some* $a \in \mathsf{I}_{\mathcal{K}}$ then return false
    for $j = 1$ to $m$ do
        guess $\{s, s'\} \subseteq Q_{\mathcal{R}}$ and $\{\gamma, \gamma'\} \subseteq \Gamma^*_{\mathcal{R}}$ with $|\gamma| \leq \mathsf{d}_{\mathcal{R}}$ and $|\gamma'| \leq \mathsf{d}_{\mathcal{R}}$
        if $\langle \mathsf{state}[j], S, \mathsf{stack}[j] \rangle \not\vdash \langle s, \epsilon, \gamma \rangle$ then return false
        if $\mathcal{L}(\mathsf{sfa}(D)) \cap \mathcal{L}_{\mathsf{d}_{\mathcal{R}}}(\mathsf{pda}(s, \gamma, s', \gamma')) = \emptyset$ then return false
        set state$[j] := s'$ and stack$[j] := \gamma'$
    set concept := $D$
if concept $\neq B$ then return false
if *there exists an index* $j \in [1..m]$ *such that* state$[j] \neq s'_j$ *or* stack$[j] \neq \gamma'_j$ then
    return false
return true
```

By the definition of $\mathsf{d}_{\mathcal{R}}$, we have that $\mathsf{d}_{\mathcal{R}}$ is linearly bounded by the number of axioms occurring in $\mathcal{R}$; hence, we need at most $O(m \cdot |\mathcal{R}|)$ space to store the two arrays. Furthermore, we need at most $O(m \cdot |\mathcal{K}|)$ space to store the counter k using binary encoding. By Definition 20, the size of $\mathsf{sfa}(D)$ is polynomial in $|\mathcal{K}|$; by Definition 4, the size of $\mathsf{pda}(s, \gamma, s', \gamma')$ is polynomial in $|\mathcal{R}|$. Overall, the space needed to store the required information is polynomial in $m \cdot |\mathcal{K}|$. Finally, following Krötzsch (2011), we can realise the check in step 9 in polynomial time. Thus, exist can be implemented so that it uses space polynomial in $m \cdot |\mathcal{K}|$.

Next, assume that the RBox $\mathcal{R}$ is fixed. Then $\mathsf{d}_{\mathcal{R}}$, $Q_{\mathcal{R}}$, $\Sigma_{\mathcal{R}}$, and $\Gamma_{\mathcal{R}}$ are all fixed as well; moreover, m is bounded by the size of $\mathcal{R}$ and so it is fixed, and M is linear in the size of $\mathcal{T}$ and $\mathcal{A}$. Thus, the number of alternatives in the nondeterministic step in line 3 of Algorithm 2 is fixed, so lines 1–5 require time polynomial in $|\mathcal{T}| + |\mathcal{A}|$. Furthermore, instead of guessing k using the nondeterministic step 6, we can repeat lines 7–15 for each $k \in [1..M]$, which requires a linear number of iterations. To show that lines 7–15 can also be implemented to run in polynomial time, we first define three sets which can be used to perform the checks in lines 9, 12, and 13.

$$\{\langle S, C, D\rangle \in \Sigma_{\mathcal{R}} \times (\mathsf{C}_{\mathcal{K}} \cup \mathsf{N}_{\mathcal{K}} \cup \{\top_c\})^2 \mid \mathcal{K} \models C \sqsubseteq \exists S.D \text{ and } \forall a \in \mathsf{I}_{\mathcal{K}} : \mathcal{K} \not\models D \sqsubseteq \{a\}\} \quad (35)$$

$$\{\langle S, \mathsf{pda}(s, \gamma, s', \gamma')\rangle \mid \text{for } S \in \Sigma_{\mathcal{R}} \text{ with } \langle s, S, \gamma\rangle \vdash \langle s', \epsilon, \gamma'\rangle\} \quad (36)$$

$$\{\langle C, \mathsf{pda}(s, \gamma, s', \gamma')\rangle \mid \text{for } C \in \mathsf{C}_{\mathcal{K}} \cup \{\top_c\} \text{ with } \mathcal{L}(\mathsf{sfa}(C)) \cap \mathcal{L}_{\mathsf{d}_{\mathcal{R}}}(\mathsf{pda}(s, \gamma, s', \gamma')) \neq \emptyset\} \quad (37)$$

Given that $\mathcal{R}$ is fixed, these sets can be computed in time polynomial in the size of $\mathcal{T}$ and $\mathcal{A}$. We next show that we can implement the for-loop in steps 7–15 to use space logarithmic in the size of $\mathcal{T}$, $\mathcal{A}$, and of the sets in equations (35)–(37). For the space usage in lines 7–15, at each computation step of the for-loop we store the information from points (a)–(f) above. Since $\mathcal{R}$ and m are fixed, however, points (a)–(c) require constant space. Furthermore, the checks in lines 9, 12, and 13 can be performed by a lookup in sets (35)–(37); by storing these sets using a suitable binary encoding and by using a binary index into the sets, this check can be implemented using logarithmic space. Finally, as $\mathsf{C}_{\mathcal{K}}$ and M are linear in the size of $\mathcal{T}$ and $\mathcal{A}$, we can store counter k, concepts D and $\mathsf{concept}$, and role S using a binary encoding, so the overall space the function needs to store is logarithmic in $|\mathcal{T}| + |\mathcal{A}|$, and the size of the sets (35)–(37). Thus, steps 7–15 require nondeterministic logarithmic space, and it is well known that this implies that steps 7–15 can be implemented to run in polynomial time. Finally, steps 16–19 clearly require polynomial time. Consequently, function exist can be implemented so that it runs in time polynomial in $|\mathcal{T}| + |\mathcal{A}|$ for fixed $\mathcal{R}$. □

We are now ready to establish the complexity of function $\mathsf{entails}(\mathcal{K}, q)$; in Section 5 we shall show that our function is worst-case optimal in combined and data complexities.

Theorem 23. *For q a BCQ over $\mathcal{K}$, function* $\mathsf{entails}(\mathcal{K}, q)$ *can be implemented so that*

1. *it uses space polynomial in the input size,*
2. *if the RBox $\mathcal{R}$ is fixed, it runs in nondeterministic polynomial time in the size of the TBox $\mathcal{T}$, the ABox $\mathcal{A}$, and the query q, and*
3. *if the RBox $\mathcal{R}$ and the query q are fixed, it runs in (deterministic) polynomial time in the size of the TBox $\mathcal{T}$ and the ABox $\mathcal{A}$.*

Proof. Let $q = \exists \vec{y}.\psi(\vec{y})$ be a Boolean CQ over $\mathcal{K}$.

As shown in Proposition 2, one can compute in lines 14 and 16 a PDA accepting language $\mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(i_S,\ \bot,\ s_0,\ \gamma_0))$ in polynomial time, so the checks in lines 14 and 16 require time (and therefore space) polynomial in $|\mathcal{K}|$ (Hopcroft et al., 2003, ch. 7). Moreover, the checks in lines 1 and 4 also require time polynomial in $|\mathcal{K}|$ (Krötzsch, 2011).

For (1), please observe that the function $\mathsf{entails}$ as specified in Algorithm 1 stores the following information at each computation step:

- a substitution σ with $\mathsf{dom}(\sigma) = \vec{y}$ and $\mathsf{rng}(\sigma) \subseteq \vec{y} \cup \mathsf{I}_\mathcal{K}$;
- a skeleton $\mathcal{S} = \langle V, E, \kappa \rangle$ for $\sigma(q)$;
- a path $v_0, \ldots, v_n$ in $\mathcal{S}$, a sequence of states $s_0, \ldots, s_n$ in $Q_\mathcal{R}$, and a sequence of words $\gamma_0, \ldots, \gamma_n$ in $\Gamma_\mathcal{R}^*$ such that $|\gamma_i| \leq \mathsf{d}_\mathcal{R}$ for each $i \in [0..n]$;
- a function L mapping each edge $\langle v, v' \rangle \in E$ to a set of generalised PDA; and,
- a walking automaton $\mathsf{wfa}(\kappa(t),\ \kappa(v_0))$ and a stationary automaton $\mathsf{sfa}(\kappa(v_0))$.

By the definition of skeleton for $\sigma(q)$, we need space polynomial in the size q and $\mathcal{K}$ to store $\mathcal{S}$. Moreover, the length of the longest path in $\mathcal{S}$ is given by the number of variables occurring in $\sigma(q)$, so we can store the sequences of vertices, states, and words in space polynomial in $|q|$ and $|\mathcal{K}|$ as well. Also, each set $L(v, v')$ contains at most m PDAs, where m is the number of binary atoms occurring in $\sigma(q)$. Then, by Lemma 22, $\mathsf{entails}$ can be implemented so that it uses space polynomial in the input size.

For (2), assume that the RBox $\mathcal{R}$ is fixed. By Lemma 22, for a fixed RBox $\mathcal{R}$, step 18 can be implemented so that it runs in time polynomial in $|\mathcal{T}| + |\mathcal{A}|$. Clearly, all other steps in Algorithm 1 can be implemented to run in nondeterministic polynomial time in the size of TBox $\mathcal{T}$, ABox $\mathcal{A}$, and query q. Consequently, for a fixed RBox $\mathcal{R}$, function $\mathsf{entails}$ can be implemented so that it runs in nondeterministic polynomial time in in the size of TBox $\mathcal{T}$, ABox $\mathcal{A}$, and query q.

For (3), assume that the RBox $\mathcal{R}$ and the query q are fixed. Then $\mathsf{d}_\mathcal{R}$, $Q_\mathcal{R}$, $\Sigma_\mathcal{R}$, and $\Gamma_\mathcal{R}$ are all fixed as well. Given that the number of variables occurring in q is fixed, the number of guessing steps required in steps 2 and 3 is fixed; also, the number of alternatives for these steps is linear in $|\mathcal{T}| + |\mathcal{A}|$. Thus, steps 2 and 3 require polynomial time. Furthermore, the maximum number of iterations of the for-loop in steps 6–16 is fixed and the length of the longest path in $\mathcal{S}$ is fixed. Thus, the number of guessing steps in lines 11 and 12 is also fixed. In addition, the number of alternatives for the guessing steps in lines 8, 11, and 12 is fixed as well. Therefore, steps 6–16 require time polynomial in $|\mathcal{T}| + |\mathcal{A}|$. Finally, since the query is fixed, the maximum number of iterations of the for-loop in steps 17 and 18 is also fixed and so, by Lemma 22, steps 17 and 18 require time polynomial in $\mathcal{T}$ and $\mathcal{A}$. Therefore, $\mathsf{entails}$ can be implemented so that it runs in time polynomial in $|\mathcal{T}| + |\mathcal{A}|$ for fixed $\mathcal{R}$ and q. $\square$

4.4 Proof of Theorem 21

We now prove that our function $\mathsf{entails}(\mathcal{K}, q)$ indeed decides $\mathcal{K} \models q$. Towards this goal, we start by proving the correctness of function exist, after which we introduce the universal interpretation of $\mathcal{K}$, and, finally, we show that $\mathsf{entails}$ is sound and complete.

4.4.1 Correctness of exist

The following proposition proves the correctness of function exist from Algorithm 2.

Lemma 24. *Function* $\mathsf{exist}(A,\, B,\, \{\mathcal{P}_j = \mathsf{pda}(s_j,\, \gamma_j,\, s'_j,\, \gamma'_j) \mid 0 \leq j \leq m\})$ *returns true if and only if there exist a natural number* $k \geq 1$*, roles* $S_1, \ldots, S_k$*, basic concepts* $A_0, \ldots, A_k$ *with* $A_0 = A$ *and* $A_k = B$*, and role chains* $\{\chi_{j,i} \mid j \in [1..m] \text{ and } i \in [1..k]\}$ *such that the following conditions hold for each* $i \in [1..k]$ *and each* $j \in [1..m]$*.*

1. *For each* $a \in \mathsf{I}_\mathcal{K}$*, we have* $\mathcal{K} \models A_{i-1} \sqsubseteq \exists S_i.A_i$ *and* $\mathcal{K} \not\models A_i \sqsubseteq \{a\}$*.*
2. *For each role* T *occurring in* $\chi_{j,i}$*, we have* $\mathcal{K} \models A_i \sqsubseteq \exists T.\mathsf{Self}$ *or* $\mathcal{K} \models \epsilon \sqsubseteq T$*.*
3. $S_1 \cdot \chi_{j,1} \cdots S_k \cdot \chi_{j,k} \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathcal{P}_j)$*.*

Proof. Consider arbitrary A, B, and $\mathcal{P}_j = \mathsf{pda}(s_j,\, \gamma_j,\, s'_j,\, \gamma'_j)$ as stated in the lemma. Moreover, let $\vdash$ be the derivation relation corresponding to $\delta_\mathcal{R}$.

($\Rightarrow$) Assume that there is a nondeterministic computation of exist such that the function returns true. Let $k \in \mathbb{N}$ be as guessed in step 6; we show that the for-loop in steps 7–15 satisfies the following invariant: after each iteration r, there exist roles $S_1, \ldots, S_r$, basic concepts $A_0, \ldots, A_r$, and, for each $j \in [1..m]$, role chains $\chi_{j,1}, \ldots, \chi_{j,r}$ such that $A_0 = A$, $A_r = \mathsf{concept}$, and the following holds for each $i \in [1..r]$ and $j \in [1..m]$.

(i) $\mathcal{K} \models A_{i-1} \sqsubseteq \exists S_i.A_i$ and, for each $a \in \mathsf{I}_\mathcal{K}$, we have $\mathcal{K} \not\models A_i \sqsubseteq \{a\}$.

(ii) For each role T occurring in $\chi_{j,i}$, we have $\mathcal{K} \models A_i \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$.

(iii) $S_1 \cdot \chi_{j,1} \cdots S_r \cdot \chi_{j,r} \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(s_j,\, \gamma_j,\, \mathsf{state}[j],\, \mathsf{stack}[j]))$.

Base case. Before the first iteration of the loop (i.e., after steps 1–5 and for $r = 0$), we have $\mathsf{concept} = A$, and $\epsilon \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(s_j,\, \gamma_j,\, \mathsf{state}[j], \mathsf{stack}[j])$ for each $j \in [1..m]$, so properties (*i*)–(*iii*) clearly hold.

Inductive step. Consider an arbitrary iteration $r \in [1..k-1]$ and assume that properties (*i*)–(*iii*) hold at the end of iteration r; we show that the same is true after iteration $r+1$. By the inductive hypothesis, there exist roles $S_1, \ldots, S_r$, basic concepts $A_0, \ldots, A_r$, and, for each $j \in [1..m]$, role chains $\chi_{j,1}, \ldots, \chi_{j,r}$ such that $A_0 = A$, $A_r = \mathsf{concept}$ and properties (*i*)–(*iii*) hold. Let role $S_{r+1} = S$ and atomic concept $A_{r+1} = D$ be as guessed in step 8. Clearly, we have $\mathcal{K} \models \mathsf{concept} \sqsubseteq \exists S_{r+1}.A_{r+1}$ and that $\mathcal{K} \not\models A_{r+1} \sqsubseteq \{a\}$ for each $a \in \mathsf{I}_\mathcal{K}$, as required for property (*i*). Furthermore, consider an arbitrary $j \in [1..m]$, let s, s', γ, and γ' be as guessed in step 11, and let $\mathsf{state}[j]$ and $\mathsf{stack}[j]$ be as at the end of iteration r; then, $\langle \mathsf{state}[j],\, S_{r+1},\, \mathsf{stack}[j] \rangle \vdash \langle s,\, \epsilon,\, \gamma \rangle$ due to step 12; furthermore, due to step 13, a role chain $\chi_{j,r+1}$ exists such that $\chi_{j,r+1} \in \mathcal{L}(\mathsf{sfa}(D)) \cap \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(s, \gamma, s', \gamma'))$. By Definition 20 of stationary automata, for each role T occurring in $\chi_{j,r+1}$ we have $\mathcal{K} \models D \sqsubseteq \exists T.\mathsf{Self}$ or

$\mathcal{K} \models \epsilon \sqsubseteq T$, as required for property (*ii*). Finally, let $\mathsf{state}[j]$ and $\mathsf{stack}[j]$ be as specified in step 14; then $S_1 \cdot \chi_{j,1} \cdots S_r \cdot \chi_{j,r} \cdot S_{r+1} \cdot \chi_{j,r+1} \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(s_j,\ \gamma_j,\ \mathsf{state}[j],\ \mathsf{stack}[j]))$, so property (*iii*) holds.

Step 16 ensures that $\mathsf{concept} = B$; furthermore, steps 17–18 ensure that $\mathsf{state}[j] = s'_j$ and $\mathsf{stack}[j] = \gamma'_j$ for each $j \in [1..m]$, so PDA $\mathcal{P}_j$ accepts $S_1 \cdot \chi_{j,1} \cdots S_k \cdot \chi_{j,k}$. Thus, properties (1)–(3) of this lemma hold, as required.

($\Leftarrow$) Let $S_1, \dots, S_n$ be roles, let $A_0, \dots, A_n$ be basic concepts with $A_0 = A$ and $A_n = B$, and let $\chi_{j,i}$ be role chains satisfying properties (1)–(3) of this lemma. Each derivation for $S_1 \cdot \chi_{j,1} \cdots S_n \cdot \chi_{j,n}$ of PDA $\mathcal{P}_j$ is of the following form, where $s_{j,n+1} = s'_j$ and $\gamma_{j,n+1} = \gamma'_j$:

$$\begin{aligned}
& & \langle s_j,\ S_1 \cdot \chi_{j,1} \cdots S_n \cdot \chi_{j,n},\ \gamma_j\rangle \vdash^* \quad & (38)\\
\langle s_{j,1},\ S_1 \cdot \chi_{j,1} \cdots S_n \cdot \chi_{j,n},\ \gamma_{j,1}\rangle \vdash & & \langle s'_{j,1},\ \chi_{j,1} \cdot S_2 \cdots S_n \cdot \chi_{j,n},\ \gamma'_{j,1}\rangle \vdash^* \quad & (39)\\
\langle s_{j,2},\ S_2 \cdot \chi_{j,2} \cdots S_n \cdot \chi_{j,n},\ \gamma_{j,2}\rangle \vdash & & \ldots \vdash^* \quad & (40)\\
\langle s_{j,i},\ S_i \cdot \chi_{j,i} \cdots S_n \cdot \chi_{j,n},\ \gamma_{j,i}\rangle \vdash & & \langle s'_{j,i},\ \chi_{j,i} \cdot S_{i+1} \cdots S_n \cdot \chi_{j,n},\ \gamma'_{j,i}\rangle \vdash^* \quad & (41)\\
\langle s_{j,i+1},\ S_{i+1} \cdot \chi_{j,i} \cdots S_n \cdot \chi_{j,n},\ \gamma_{j,i+1}\rangle \vdash & & \ldots \vdash^* \quad & (42)\\
\langle s_{j,n},\ S_n \cdot \chi_{j,n},\ \gamma_{j,n}\rangle \vdash & & \langle s'_{j,n},\ \chi_{j,n},\ \gamma'_{j,n}\rangle \vdash^* \quad & (43)\\
\langle s_{j,n+1},\ \epsilon,\ \gamma_{j,n+1}\rangle & & & (44)
\end{aligned}$$

Transition from (38) to (39) is 'special' in the sense that it allows $\mathcal{P}_j$ to make an arbitrary number of ε-transitions; the rest of the derivation is regular and consists of reading S_i and $\chi_{j,i}$. Thus, $s_{j,i}$ and $s'_{j,i}$ are the states of $\mathcal{P}_j$ before and after, respectively, reading S_i, and $\gamma_{j,i}$ and $\gamma'_{j,i}$ are the respective stacks. By property (3) of this lemma, we have $|\gamma_{j,i}| \leq \mathsf{d}_\mathcal{R}$ and $|\gamma'_{j,i}| \leq \mathsf{d}_\mathcal{R}$.

Let $X_i = \langle A_i, s_{1,i}, \gamma_{1,i}, \dots, s_{m,i}, \gamma_{m,i}\rangle$. For each PDA, there are $|Q_\mathcal{R}|$ many different states, $1 + |\mathsf{C}_\mathcal{K}|$ different elements in $\{\top_c\} \cup \mathsf{C}_\mathcal{K}$; furthermore, there are $\sum_{\ell=0}^{\mathsf{d}_\mathcal{R}} |\Gamma_\mathcal{R}|^\ell$ many different stacks of length at most $\mathsf{d}_\mathcal{R}$. As $|\Gamma_\mathcal{R}| > 0$ and $\mathsf{d}_\mathcal{R} > 0$, we have $\sum_{\ell=0}^{\mathsf{d}_\mathcal{R}} |\Gamma_\mathcal{R}|^\ell \leq |\Gamma_\mathcal{R}|^{1+\mathsf{d}_\mathcal{R}}$; consequently, there are at most M distinct such tuples. Thus, for some $k \leq M$, we have that $X_k = X_{n+1}$. But then, $A_k = B$; furthermore, for each $j \in [1..m]$, we have $s_{j,k} = s_{j,n+1} = s'_j$ and $\gamma_{j,k} = \gamma_{j,n+1} = \gamma'_j$, so we have $S_1 \cdot \chi_{j,1} \cdots S_k \cdot \chi_{j,k} \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathcal{P}_j)$.

We can now easily construct a nondeterministic computation of exist as follows. In step 3, for each j we let $s = s_{j,1}$ and $\gamma = \gamma_{j,1}$; clearly, condition in step 4 is not satisfied. For each r in the for-loop in lines 7–15, we proceed as follows.

- In step 8 we let $S = S_i$ and $D = A_i$, respectively; clearly, condition in step 9 is not satisfied due to property (1).
- For each $j \in [1..m]$, we let $s = s'_{j,r}$, $s' = s_{j,r+1}$, $\gamma = \gamma'_{j,r}$, and $\gamma' = \gamma_{j,r+1}$; clearly, condition in step 12 is not satisfied due to the form of the derivation; furthermore, condition in step 13 is not satisfied due to property (2) and Definition 20.

Finally, conditions in steps 16 and 17 are not satisfied due to the way in which we chose k. Therefore, function exist returns true in step 19. □

Rule	Precondition	Conclusion
(cr1)	$\mathcal{K} \models A_1 \sqcap A_2 \sqsubseteq B$ $\{A_1(w),\, A_2(w)\} \subseteq I$	$B(w)$
(cr2)	$\mathcal{K} \models A \sqsubseteq \exists S.B$ $\mathcal{K} \models B \sqsubseteq \{a\}$ for some $a \in \mathsf{I}_{\mathcal{K}}$ $A(w) \in I$	$S(w, a),\, B(a)$
(cr3)	$\mathcal{K} \models A \sqsubseteq \exists S.B$ $\mathcal{K} \not\models B \sqsubseteq \{a\}$ for each $a \in \mathsf{I}_{\mathcal{K}}$ $A(w) \in I$	$S(w, f_{S,B}(w)),\, B(f_{S,B}(w))$ $\top_c(f_{S,B}(w)),\, \top_r(f_{S,B}(w), f_{S,B}(w))$ $\top_r(f_{S,B}(w), w')$ for each term w' occurring in I $\top_r(w', f_{S,B}(w))$ for each term w' occurring in I
(cr4)	$\mathcal{K} \models \exists S.A \sqsubseteq B$ $\{S(w, w'),\, A(w')\} \subseteq I$	$B(w)$
(cr5)	$\mathcal{K} \models \exists S.\mathsf{Self} \sqsubseteq B$ $S(w, w) \in I$ role S is simple	$B(w)$
(cr6)	$\mathcal{K} \models A \sqsubseteq \exists S.\mathsf{Self}$ $A(w) \in I$ role S is simple	$S(w, w)$
(cr7)	$\rho \in \mathcal{L}(S)$ $\rho(w, w') \in I$	$S(w, w')$

Table 5: Rules of the consequence-based chase

4.4.2 Consequence-Based Chase and Universal Interpretations

To prove that $\mathsf{entails}(\mathcal{K}, q)$ is sound and complete, we interpret $\mathcal{K}$ using the forest-shaped universal interpretation described in Section 4.2. Towards this goal, we next define some auxiliary notions, then define the universal interpretation, and, finally, we prove two properties of such interpretation.

The *universe* of $\mathcal{K}$ is the set of all terms built from the individuals occurring in $\mathcal{K}$ and the unary function symbols of the form $f_{S,A}$ with $S \in \Sigma_{\mathcal{R}}$ and $A \in \mathsf{C}_{\mathcal{K}}$. Since $\mathcal{K}$ is normalised, the universe of $\mathcal{K}$ is nonempty. A *fact* is a ground atom constructed using the predicates occurring in $\mathcal{K}$ and the terms from the universe of $\mathcal{K}$. For a role chain $\rho = S_1 \cdots S_n$, terms w and w', and a set of facts I, we write $\rho(w, w') \in I$ if (not necessarily distinct) terms $w = w_0, \ldots, w_n = w'$ exists such that $S_i(w_{i-1}, w_i) \in I$ for each $i \in [1..n]$. A set of facts I *entails* a Boolean CQ $q = \exists \vec{y}.\, \psi(\vec{y})$, written $I \models q$, if a substitution τ exists such that $\mathsf{dom}(\tau) = \vec{y}$ and $\tau(q) \subseteq I$. The *universal interpretation* $\mathcal{I}_{\mathcal{K}}$ of $\mathcal{K}$ is defined as follows.

Definition 25. *A* chase rule *from Table 5 is* applicable *to a set of facts I if the preconditions of the rule are satisfied, but I does not contain all conclusions of the rule. A* consequence-based chase *(often just* chase*) for $\mathcal{K}$ is a sequence of sets of facts $I_0, I_1, \ldots$ where*

$$I_0 = \{\{a\}(a),\, \top_c(a),\, \top_r(a, b) \mid \{a, b\} \subseteq \mathsf{I}_{\mathcal{K}}\} \cup \mathcal{A} \tag{45}$$

and, for each $i \geq 1$, set I_{i+1} is obtained by extending I_i with the conclusion of one (arbitrarily chosen) chase rule applicable to I_i, and $I_{i+1} = I_i$ if no chase rule is applicable to I_i. This *sequence must be* fair*—that is, if a derivation rule is applicable to some I_i for a specific precondition, then $j \geq i$ exists such that I_{j+1} is obtained from I_j by applying the rule to the mentioned precondition. Set $\mathcal{I}_{\mathcal{K}} = \bigcup_{i \in \mathbb{N}} I_i$ is a* universal interpretation *of $\mathcal{K}$.*

Since $\mathcal{K}$ is in normal form, we have $\mathcal{K} \not\models \{a\} \sqsubseteq \{b\}$ for all distinct individuals a and b in $\mathsf{I}_{\mathcal{K}}$; hence, at most one individual $a \in \mathsf{I}_{\mathcal{K}}$ exists in rule (cr2) such that $\mathcal{K} \models B \sqsubseteq \{a\}$. Because of that, it is straightforward to see that $\mathcal{I}_{\mathcal{K}}$ is independent from the order in which the chase rules are applied, so we call $\mathcal{I}_{\mathcal{K}}$ *the* universal interpretation of $\mathcal{K}$. Moreover, due to fairness, no derivation rule is applicable to $\mathcal{I}_{\mathcal{K}}$—that is, for each chase rule from Table 5 either the preconditions of the rule are not satisfied in $\mathcal{I}_{\mathcal{K}}$, or $\mathcal{I}_{\mathcal{K}}$ contains all the conclusions of the rule. Finally, it is well-known that, if $\mathcal{K}$ is consistent, then $\mathcal{I}_{\mathcal{K}}$ can be homomorphically embedded into any model of $\mathcal{K}$ (Krötzsch et al., 2007). Consequently, the universal interpretation $\mathcal{I}_{\mathcal{K}}$ can be used to answer arbitrary Boolean CQs over $\mathcal{K}$.

Fact 26. *For each Boolean CQ q, we have $\mathcal{K} \models q$ if and only if $\mathcal{K} \models \top_c \sqsubseteq \bot_c$ or $\mathcal{I}_{\mathcal{K}} \models q$.*

Next, we show how $\mathcal{I}_{\mathcal{K}}$ relates to the axioms entailed by $\mathcal{K}$. To this end, let ι be the following function mapping each term w in the universe of $\mathcal{K}$ to a basic concept:

$$\iota(w) := \begin{cases} \{w\} & \text{if } w \in \mathsf{I} \\ A & \text{if } w \text{ is of the form } w = f_{S,A}(w') \end{cases}$$

Proposition 27. *The universal model $\mathcal{I}_{\mathcal{K}}$ satisfies the following properties.*

1. *For each $A(w) \in \mathcal{I}_{\mathcal{K}}$, we have $\mathcal{K} \models \iota(w) \sqsubseteq A$.*
2. *For each $S(w, w') \in \mathcal{I}_{\mathcal{K}}$, a nonempty role chain $\rho = \chi_0 \cdot S_1 \cdot \chi_1 \cdots \chi_{m-1} \cdot S_m \cdot \chi_m$ with $\rho \in \mathcal{L}(S)$ and terms $w_0, \ldots, w_m$ from the universe of $\mathcal{K}$ with $w_0 = w$ and $w_m = w'$ exist such that*
   (a) *for each $i \in [1..m]$, either $w_i \in \mathsf{I}_{\mathcal{K}}$, or an atomic concept $A_i \in \{\top_c\} \cup \mathsf{C}_{\mathcal{K}}$ exists such that $w_i = f_{S_i,A_i}(w_{i-1})$ and $\mathcal{K} \not\models A_i \sqsubseteq \{a\}$ for each individual $a \in \mathsf{I}_{\mathcal{K}}$,*
   (b) *for each $i \in [1..m]$, we have $\mathcal{K} \models \iota(w_{i-1}) \sqsubseteq \exists S_i.\iota(w_i)$, and*
   (c) *for each $i \in [0..m]$ and each role T occurring in χ_i, we have $\mathcal{K} \models \iota(w_i) \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$.*

Proof. Let $I_0, I_1, \ldots$ be a chase sequence for $\mathcal{K}$. We show by induction on rule applications that properties (1) and (2) are satisfied for each $A(w) \in I_n$ and each $S(w, w') \in I_n$, respectively, and that I_n additionally satisfies the following property:

3. For each term w occurring in I_n, we have $\mathcal{K} \models \exists x.\iota(w)(x)$.

By the definition of I_0 and (cr3), for each I_n and all terms w and w' occurring in I_n, we clearly have $\{\top_c(w), \top_r(w, w'), \top_r(w', w)\} \subseteq I_n$.

Base case. Consider I_0, and note that each term w occurring in I_0 is an individual so $\iota(w) = \{w\}$. Consider some $A(a) \in I_0$; then either $A(a) \in \mathcal{A}$, $A = \{a\}$, or $A = \top_c$, so we have $\mathcal{K} \models \{a\} \sqsubseteq A$, and property (1) holds. Furthermore, consider some $S(a, b) \in I_0$; then

$S(a,b) \in \mathcal{A}$ or $S = \top_r$, so we have $\mathcal{K} \models \{a\} \sqsubseteq \exists S.\{b\}$, and property (2) holds for $w_0 = a$, $w_1 = b$, and $\rho = S$. Finally, property (3) holds because $\mathcal{K} \models \exists x.\{a\}(x)$ for each $a \in \mathsf{I}_{\mathcal{K}}$.

Inductive step. Assume that some I_n satisfies properties (1)–(3). By considering each derivation rule, we assume that the rule is applicable to I_n as shown in Table 5, and we show that properties (1)–(3) hold for all conclusions of the rule. Note that only rule (cr3) can affect property (3), and we do not explicitly consider properties that hold vacuously.

(cr1) By the inductive hypothesis, we have $\mathcal{K} \models \iota(w) \sqsubseteq A_1$ and $\mathcal{K} \models \iota(w) \sqsubseteq A_2$, which implies $\mathcal{K} \models \iota(w) \sqsubseteq B$, as required for property (1).

(cr2) By the inductive hypothesis, we have $\mathcal{K} \models \iota(w) \sqsubseteq A$ and $\mathcal{K} \models \exists x.\iota(w)(x)$, which clearly imply $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.B$ and $\mathcal{K} \models \exists x.B(x)$. Moreover, $\iota(a) = \{a\}$, so $\mathcal{K} \models \iota(a) \sqsubseteq B$, and property (1) holds. Finally, since $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.\iota(a)$, property (2) holds for $w_0 = w$, $w_1 = a$, and $\rho = S$.

(cr3) Let $w'' = f_{S,B}(w)$. By the inductive hypothesis, we have $\mathcal{K} \models \iota(w) \sqsubseteq A$ and $\mathcal{K} \models \exists x.\iota(w)(x)$, so we have $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.B$ and $\mathcal{K} \models \exists x.B(x)$. Moreover, $\iota(w'') = B$, so $\mathcal{K} \models \iota(w'') \sqsubseteq B$, $\mathcal{K} \models \iota(w'') \sqsubseteq \top_c$, and $\mathcal{K} \models \exists x.\iota(w'')(x)$, as required for properties (1) and (3), respectively. For property (2), we consider all role assertions derived by the rule.

- $S(w, w'')$. Note that $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.\iota(w'')$ and $\mathcal{K} \not\models B \sqsubseteq \{a\}$ for each $a \in \mathsf{I}_{\mathcal{K}}$, so property (2) holds for $w_0 = w$, $w_1 = w''$, and $\rho = S$.

- $\top_r(w'', w'')$. Clearly, property (2) holds for $w_0 = w''$ and $\rho = \chi_0 = \top_r$.

- $\top_r(w'', w')$ for some term w' occurring in I_n. Let a be the individual that w' is rooted in; then, $\top_r(a, w') \in I_n$ so, by the inductive hypothesis, a role chain $\rho \in \mathcal{L}(\top_r)$ and terms $a = w_0, \ldots, w_m = w'$ exist satisfying properties (a)–(c). Since $\mathcal{K} \models \exists x.\iota(w'')(x)$, we have $\mathcal{K} \models \iota(w'') \sqsubseteq \exists \top_r.\{a\}$; thus, $\top_r \cdot \rho$ and $w'', w_0, \ldots, w_m$ satisfy property (2).

- $\top_r(w', w'')$ for some term w' occurring in I_n. Then, $\top_r(w', w) \in I_n$ so, by the inductive hypothesis, a role chain $\rho \in \mathcal{L}(\top_r)$ and terms $w' = w_0, \ldots, w_m = w$ exist satisfying properties (a)–(c). But then, $\rho \cdot \top_r$ and $w_0, \ldots, w_m, w''$ satisfy property (2).

(cr4) By the inductive hypothesis, we have $\mathcal{K} \models \iota(w') \sqsubseteq A$; moreover, terms $w_0, \ldots, w_m$ with $w_0 = w$ and $w_m = w'$ and a nonempty role chain $\rho = \chi_0 \cdot S_1 \cdot \chi_1 \cdots \chi_{m-1} \cdot S_m \cdot \chi_m$ with $\rho \in \mathcal{L}(S)$ exist satisfying properties (a)–(c). By the definition of $\mathcal{L}(S)$, we have $\mathcal{K} \models \rho \sqsubseteq S$; together with the entailments in properties (b) and (c), we have $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.\iota(w')$. But then, we have $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.A$, which implies $\mathcal{K} \models \iota(w) \sqsubseteq B$, as required for property (1).

(cr5) By the inductive hypothesis, a nonempty role chain $\rho \in \mathcal{L}(S)$ exist satisfying properties (a)–(c); moreover, $\mathcal{K} \models \rho \sqsubseteq S$ by the definition of $\mathcal{L}(S)$. Role S is simple, so $|\rho| = 1$, and therefore ρ can have one of the following two forms.

- $\rho = \chi_0$ and $\chi_0 = T$. By property (c), we have $\mathcal{K} \models \iota(w) \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$. Furthermore, due to $\mathcal{K} \models \rho \sqsubseteq S$, we have $\mathcal{K} \models T \sqsubseteq S$. But then, $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.\mathsf{Self}$, and so $\mathcal{K} \models \iota(w) \sqsubseteq B$ holds, as required for property (1).

- $\rho = S_1$. Terms w_0 and w_1 satisfying property (2) are then both equal to w; moreover, w_1 is not of the form $f_{S_1,A_1}(w)$, so $w \in \mathsf{I}_{\mathcal{K}}$. Furthermore, by property (b) we have

$\mathcal{K} \models \iota(w) \sqsubseteq \exists S_1.\iota(w)$; together with $w \in \mathsf{I}_\mathcal{K}$, we have $\mathcal{K} \models \iota(w) \sqsubseteq \exists S_1.\mathsf{Self}$. Finally, due to $\mathcal{K} \models \rho \sqsubseteq S$, we have $\mathcal{K} \models S_1 \sqsubseteq S$. But then, $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.\mathsf{Self}$, and so $\mathcal{K} \models \iota(w) \sqsubseteq B$ holds, as required for property (1).

(cr6) By the inductive hypothesis, we have $\mathcal{K} \models \iota(w) \sqsubseteq A$, from which we can conclude $\mathcal{K} \models \iota(w) \sqsubseteq \exists S.\mathsf{Self}$, so property (2) holds for $w_0 = w$ and $\rho = \chi_0 = S$.

(cr7) If $\rho = \epsilon$, then $w = w'$ and $\mathcal{K} \models \epsilon \sqsubseteq S$, and property (2) holds for $w_0 = w$ and $\rho = \chi_0 = S$. Otherwise, assume that ρ is nonempty and of the form $\rho = S_1 \cdots S_k$. Thus, terms $w_0, \ldots, w_k$ with $w_0 = w$ and $w_k = w'$ exist such that $S_i(w_{i-1}, w_i) \in \mathcal{I}_n$ for each $i \in [1..k]$. By the inductive hypothesis, for each $i \in [1..k]$, terms $w_0^i, \ldots, w_{m_i}^i$ with $w_0^i = w_{i-1}$ and $w_{m_i}^i = w_i$ and a role chain ρ^i with $\rho^i \in \mathcal{L}(S_i)$ exist satisfying properties (a)–(c); note that $w_0^i = w_0 = w$, that $w_{m_k}^k = w_k = w'$, and that $w_{m_{i-1}}^{i-1} = w_0^i$ for each $i \in [1..k]$. By the definition of $\mathcal{L}(S)$, then $\rho^1 \cdots \rho^k \in \mathcal{L}(S)$, and so property (2) holds for role chain $\rho^1 \cdots \rho^k$ and terms $w_0, w_1^1, \ldots, w_{m_1}^1, \ldots, w_1^k, \ldots, w_{m_k}^k$. □

4.4.3 Soundness

We are now ready to show that our algorithm $\mathsf{entails}$ is sound.

Lemma 28. *If a nondeterministic computation exists such that* $\mathsf{entails}(\mathcal{K}, q)$ *returns true, then* $\mathcal{K} \models q$.

Proof. Assume that a nondeterministic computation exists such that $\mathsf{entails}(\mathcal{K}, q)$ returns true. If our algorithm returns true in step 1, then $\mathcal{K} \models q$, as $\mathcal{K}$ is inconsistent; hence, in the rest of this proof, we assume that $\mathcal{K}$ is consistent and show that $\mathcal{I}_\mathcal{K} \models q$. To this end, let substitution σ, skeleton $\mathcal{S} = \langle V, E, \kappa\rangle$, and function L be as determined by $\mathsf{entails}$. Graph $\langle V, E\rangle$ is a forest rooted in the individuals occurring in $\mathcal{K}$ so, by structural induction on this forest, we define mapping τ from V to the universe of $\mathcal{K}$ that will satisfy the following:

(i) for each $v \in V$, we have $\kappa(v)(\tau(v)) \in \mathcal{I}_\mathcal{K}$; and

(ii) for each $\langle v, v'\rangle \in E$ and each $\mathcal{P}_j \in L(v, v')$, a role chain $\rho_j \in \mathcal{L}(\mathcal{P}_j)$ exists such that $\rho_j(\tau(v), \tau(v')) \in \mathcal{I}_\mathcal{K}$.

Base case. For each $a \in \mathsf{I}_\mathcal{K}$, let $\tau(a) = a$. Since $\kappa(a) = \{a\}$ and $\{a\}(a) \in \mathcal{I}_\mathcal{K}$, the first property clearly holds, and the second property is vacuous.

Inductive step. Consider $\langle v, v'\rangle \in E$ such that $\tau(v)$ has been defined, but $\tau(v')$ has not; let $L(v, v') = \{\mathcal{P}_1, \ldots, \mathcal{P}_m\}$. Since $\mathsf{exist}(\kappa(v), \kappa(v'), L(v, v'))$ returns true, by Lemma 24 we have that roles $S_1, \ldots, S_n$, atomic concepts $A_1, \ldots, A_n$, and, for each $j \in [1..m]$, a role chain $\rho_j = S_1 \cdot \chi_{j,1} \cdots S_n \cdot \chi_{j,n}$ exist such that $n \geq 1$, $A_0 = \kappa(v)$ and $A_n = \kappa(v')$, and the following holds for each $i \in [1..n]$ and each $j \in [1..m]$.

1. For each $a \in \mathsf{I}_\mathcal{K}$, we have $\mathcal{K} \models A_{i-1} \sqsubseteq \exists S_i.A_i$ and $\mathcal{K} \not\models A_i \sqsubseteq \{a\}$.
2. For each role T occurring in $\chi_{j,i}$, we have $\mathcal{K} \models A_i \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$.
3. $\rho_j \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathcal{P}_j)$.

Let $w_0 = \tau(v)$; let $w_i = f_{S_i,A_i}(w_{i-1})$ for $i \in [1..n]$; and let $\tau(v') = w_n$. Since $A_0 = \kappa(v)$, by the inductive hypothesis we have $A_0(\tau(v)) \in \mathcal{I}_\mathcal{K}$. Furthermore, (cr3) is not applicable to $\mathcal{I}_\mathcal{K}$ so, for each $i \in [1..n]$, we have $S_i(w_{i-1}, w_i) \in \mathcal{I}_\mathcal{K}$ and $A_i(w_i) \in \mathcal{I}_\mathcal{K}$; thus, $\kappa(\tau(v')) \in \mathcal{I}_\mathcal{K}$, as required. Finally, for each role T occurring in each $\chi_{j,i}$, we have $\mathcal{K} \models A_i \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$; (cr6) and (cr7) are not applicable to $\mathcal{I}_\mathcal{K}$, respectively, so we have $T(w_i, w_i) \in \mathcal{I}_\mathcal{K}$; thus, we have $\rho_j(\tau(v), \tau(v')) \in \mathcal{I}_\mathcal{K}$, as required.

We next show that $\tau(\sigma(q)) \subseteq \mathcal{I}_\mathcal{K}$ by considering independently each atom in $\sigma(q)$. To prove the lemma, we can combine τ and σ in the obvious way.

Consider an arbitrary unary atom $A(t)$ in $\sigma(q)$. By step 4 of Algorithm 1, we have $\mathcal{K} \models \kappa(t) \sqsubseteq A$, which also implies $\mathcal{K} \models \kappa(t) \sqcap \kappa(t) \sqsubseteq A$. By property (i), we have $\kappa(t)(\tau(t)) \in \mathcal{I}_\mathcal{K}$. Since rule (cr1) is not applicable to $\mathcal{I}_\mathcal{K}$, we have $A(\tau(t)) \in \mathcal{I}_\mathcal{K}$, as required.

Consider an arbitrary binary atom $S(t, u)$ in $\sigma(q)$. Let $v_0, \ldots, v_n$, $s_0, \ldots, s_n$, and $\gamma_0, \ldots, \gamma_n$ be as determined in steps 8–11 when Algorithm 1 considers atom $S(t, u)$. For each $i \in [1..n]$, we have $\mathsf{pda}(s_{i-1}, \gamma_{i-1}, s_i, \gamma_i) \in L(v_{i-1}, v_i)$ by step 12; but then, by property (ii) a role chain ρ_i exists such that $\rho_i \in \mathcal{L}(\mathsf{pda}(s_{i-1}, \gamma_{i-1}, s_i, \gamma_i))$ and $\rho_i(\tau(v_{i-1}), \tau(v_i)) \in \mathcal{I}_\mathcal{K}$. Next, we define ρ_0 by considering the following two cases.

- $v_0 \in \mathsf{I}$. By step 14 of Algorithm 1, a role chain $\rho_0 = S_1 \cdots S_k$ exists such that $\rho_0 \in \mathcal{L}(\mathsf{wfa}(\kappa(t), \kappa(v_0)))$. By property (i), we have $\kappa(t)(\tau(t)) \in \mathcal{I}_\mathcal{K}$; moreover, by Definition 20, basic concepts $\kappa(t) = A_0, A_1, \ldots, A_k = \{v_0\}$ exist with $\mathcal{K} \models A_{j-1} \sqsubseteq \exists S_j.A_j$ for each $j \in [1..k]$. Rules (cr2) and (cr3) are not applicable to $\mathcal{I}_\mathcal{K}$, so we have $\rho_0(\tau(t), \tau(v_0)) \in \mathcal{I}_\mathcal{K}$.
- $v_0 \notin \mathsf{I}$, which implies $v_0 = t$. By step 16 of Algorithm 1, a role chain $\rho_0 = T_1 \cdots T_k$ exists such that $\rho_0 \in \mathcal{L}(\mathsf{sfa}(\kappa(v_0)))$. By property (i), we have $\kappa(v_0)(\tau(v_0)) \in \mathcal{I}_\mathcal{K}$; moreover, by Definition 20, $\mathcal{K} \models \kappa(v_0) \sqsubseteq \exists T_j.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T_j$ for each $j \in [1..k]$. Rules (cr6) and (cr7), respectively, are not applicable to $\mathcal{I}_\mathcal{K}$, thus $\rho_0(\tau(t), \tau(v_0)) \in \mathcal{I}_\mathcal{K}$.

In either case, by steps 14 and 16 we have $\rho_0 \in \mathcal{L}(\mathsf{pda}(i_S, \perp, s_0, \gamma_0))$. Now let $\rho' = \rho_0 \cdots \rho_n$; note that we can have $n = 0$, in which case $\rho' = \rho_0$. Clearly, we have $\rho'(\tau(t), \tau(v_n)) \in \mathcal{I}_\mathcal{K}$, where $v_n = u$. Moreover, $\rho' \in \mathcal{L}(\mathsf{pda}(i_S, \perp, s_n, \gamma_n))$ with $s_n = f_S$ and $\gamma_n = \perp$. Finally, rule (cr7) is not applicable to $\mathcal{I}_\mathcal{K}$, so $S(\tau(t), \tau(u)) \in \mathcal{I}_\mathcal{K}$, as required. □

4.4.4 Completeness

We next prove that our encoding is also complete, thus proving Theorem 21.

Lemma 29. *If $\mathcal{K} \models q$, then a nondeterministic computation exists such that* $\mathsf{entails}(\mathcal{K}, q)$ *returns true.*

Proof. Assume that $\mathcal{K} \models q$. If $\mathcal{K}$ is inconsistent, then $\mathsf{entails}(\mathcal{K}, q)$ returns true, as required; hence, in the rest of this proof, we assume that $\mathcal{K}$ is consistent. But then, $\mathcal{I}_\mathcal{K} \models q$, so a substitution π exists such that $\pi(q) \subseteq \mathcal{I}_\mathcal{K}$. Let ι be as defined in Section 4.4.2.

For the substitution σ in step 2, let $\sigma(y) := \pi(y)$ if $\pi(y) \in \mathsf{I}$; otherwise, let $\sigma(y)$ be an arbitrary, but fixed, variable y' from q such that $\pi(y) = \pi(y')$. It is straightforward to see that $\pi(\sigma(q)) \subseteq \mathcal{I}_\mathcal{K}$.

For the skeleton $\mathcal{S} = \langle V,\, E,\, \kappa \rangle$ in step 3, set V contains $\mathsf{I}_\mathcal{K}$ and the variables occurring in $\sigma(q)$, and $\kappa(y) = \iota(\pi(v))$ for each variable $y \in V$. Furthermore, let $\prec$ be the smallest irreflexive and transitive relation on the universe of $\mathcal{K}$ such that $w \prec f_{S,A}(w)$ for each term w in the universe of $\mathcal{K}$; then, let $\langle v, v' \rangle \in E$ if and only if $\pi(v) \prec \pi(v')$ and no $v'' \in V$ exists such that $\pi(v) \prec \pi(v'') \prec \pi(v')$. By the definition of $\prec$, graph $\langle V, E \rangle$ is a forest rooted in $\mathsf{I}_\mathcal{K}$, as required by Definition 18.

In step 4, for an arbitrary atom $A(t)$ in $\sigma(q)$, we have $A(\pi(t)) \in \mathcal{I}_\mathcal{K}$; by property (1) of Proposition 27, we have $\mathcal{K} \models \iota(\pi(t)) \sqsubseteq A$; hence, the condition is not satisfied.

Now consider an arbitrary edge $\langle v, v' \rangle \in E$; and let $w_0, \ldots, w_k$ be terms, let $A_1, \ldots, A_k$ be atomic concepts, and let $S_1, \ldots, S_k$ be roles such that $w_0 = \pi(v)$, $w_k = \pi(v')$, and $w_i = f_{S_i,A_i}(w_{i-1})$ for each $i \in [1..k]$; finally, let $A_0 = \kappa(v)$. Note that all of these are uniquely defined by the edge, and that, by the construction of $\mathcal{I}_\mathcal{K}$, for each $i \in [1..k]$, we have $\mathcal{K} \models A_{i-1} \sqsubseteq \exists S_i.A_i$ and $\mathcal{K} \not\models A_i \sqsubseteq \{a\}$ for each $a \in \mathsf{I}_\mathcal{K}$. Then, a role chain ρ is *compatible* with the edge $\langle v, v' \rangle$ if role chains $\chi_1, \ldots, \chi_k$ exists such that $\rho = S_1 \cdot \chi_1 \cdots S_k \cdot \chi_k$ and, for each $i \in [1..k]$ and each role T occurring in χ_i, we have $\mathcal{K} \models A_i \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$. In the rest of this proof we will show the following property.

($\Diamond$) For each PDA $\mathcal{P} \in L(v, v')$, a role chain $\rho \in \mathcal{L}(\mathcal{P})$ exists that is compatible with the edge $\langle v, v' \rangle$.

By Lemma 24 and the above definition of compatibility, property ($\Diamond$) implies that the condition in step 18 is not satisfied for edge $\langle v, v' \rangle$.

For the loop in steps 6–16, let $S(t, u)$ be an arbitrary binary atom in $\sigma(q)$; we next determine the required nondeterministic choices that preserve ($\Diamond$) in step 12, and that satisfy conditions in steps 14 and 16, which completes the proof of this lemma. Let $a_u \in \mathsf{I}_\mathcal{K}$ be the unique individual connected to u in $\langle V, E \rangle$. Since $S(\pi(t), \pi(u)) \in \mathcal{I}_\mathcal{K}$, a nonempty role chain $\rho = \chi_0 \cdot S_1 \cdot \chi_1 \cdots \chi_{m-1} \cdot S_m \cdot \chi_m$ with $\rho \in \mathcal{L}(S)$ and terms $w_0, \ldots, w_m$ from the universe of $\mathcal{K}$ with $w_0 = \pi(t)$ and $w_m = \pi(u)$ exist satisfying property (2) of Proposition 27. To define vertex v_0 in step 8, we consider two possibilities, and for each we also define an index $\ell_0 \in [0..m]$ such that $w_{\ell_0} = \pi(v_0)$.

- If some $j \in [0..m]$ exists such that $w_j \in \mathsf{I}_\mathcal{K}$, let $v_0 = a_u$ and let ℓ_0 be the largest index such that $w_{\ell_0} = a_u$.
- Otherwise, let $v_0 = t$ and let $\ell_0 = 0$.

Let $v_0, \ldots, v_n$ be the unique path connecting v_0 to u in $\mathcal{S}$. By the definition of ℓ_0 and the form of the terms $w_{\ell_0+1}, \ldots, w_m$, we have $w_j \not\in \mathsf{I}$ for each $j \in [\ell_0 + 1..m]$; $\pi(v_0) = w_{\ell_0}$; and $\pi(v_n) = w_m$. Thus, for each $i \in [1..n]$, a unique index ℓ_i exists such that $\pi(v_i) = w_{\ell_i}$. Now let $\rho_0 = \chi_0 \cdots S_{\ell_0} \cdot \chi_{\ell_0}$, and let $\rho_i = S_{\ell_{i-1}+1} \cdot \chi_{\ell_{i-1}+1} \cdots S_{\ell_i} \cdot \chi_{\ell_i}$ for each $i \in [1..n]$; clearly, $\rho = \rho_0 \cdots \rho_n$. By properties (*a*)–(*c*) of Proposition 27, for each $i \in [1..n]$, role chain ρ_i is compatible with the edge $\langle v_{i-1},\, v_i \rangle$. Furthermore, $\rho \in \mathcal{L}(S)$ and Theorem 8 imply $\rho \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(i_S,\, \bot,\, f_S,\, \bot))$, and so states $s_0, \ldots, s_n$ with $s_n = f_S$ and words $\gamma_0, \ldots, \gamma_n$ with $\gamma_n = \bot$ exist such that $\rho_0 \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(i_S,\, \bot,\, s_0,\, \gamma_0))$ and $\rho_i \in \mathcal{L}_{\mathsf{d}_\mathcal{R}}(\mathsf{pda}(s_{i-1},\, \gamma_{i-1},\, s_i,\, \gamma_i)$ for each $i \in [1..n]$. Since each ρ_i is compatible with $\langle v_{i-1}, v_i \rangle$, step 12 preserves property ($\Diamond$), as required. Finally, we consider step 13.

- $v_0 \in \mathsf{I}$. By property (b) of Proposition 27, $\mathcal{K} \models \iota(w_{j-1}) \sqsubseteq \exists S_j.\iota(w_j)$ for each $j \in [1..\ell_0]$. Furthermore, by property c of Proposition 27, $\mathcal{K} \models \iota(w_j) \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$ for each $j \in [0..\ell_0]$ and each role T occurring in χ_j; thus, $\mathcal{K} \models \iota(w_j) \sqsubseteq \exists T.\iota(w_j)$. But then, $\rho_0 \in \mathcal{L}(\mathsf{wfa}(\kappa(t), \kappa(v_0)))$, so condition in step 14 is not satisfied.
- $v_0 \notin \mathsf{I}$, so $v_0 = t$ and $\rho_0 = \chi_0$. By property (c) of Proposition 27, $\mathcal{K} \models \iota(w_j) \sqsubseteq \exists T.\mathsf{Self}$ or $\mathcal{K} \models \epsilon \sqsubseteq T$ for each role T occurring in χ_0. But then, $\rho_0 \in \mathcal{L}(\mathsf{sfa}(\kappa(v_0)))$, so condition in step 16 is not satisfied. □

5. The Lower Complexity Bound

In the previous section, we presented a BCQ answering algorithm for $\mathcal{ELRO}^+$ that uses space polynomial in the total size of the input. This algorithm is worst-case optimal in combined complexity since Krötzsch et al. (2007) reduced the PSpace-hard problem of checking nonemptiness of the intersection of the languages generated by m deterministic finite automata $\mathcal{F}_1 \dots \mathcal{F}_m$ over a common alphabet Σ (Kozen, 1977) to BCQ answering in $\mathcal{ELRO}^+$. In the knowledge base $\mathcal{K}$ encoding the problem, a regular RBox contains roles $S_1 \dots S_m$ such that $\mathcal{L}(S_i) = \mathcal{L}(\mathcal{F}_i)$ for each $i \in [1..m]$; furthermore, a TBox ensures that the universal interpretation $\mathcal{I}_\mathcal{K}$ is a rooted tree so, for each $\rho \in \Sigma^*$, a term w_ρ exists that is reachable from the root by a chain of roles corresponding to ρ; finally, a Boolean CQ contains m atoms that check whether $\bigcap_i \mathcal{L}(\mathcal{F}_i)$ is nonempty. We next improve this lower bound by showing that the problem is hard already in the restricted setting where the query, the TBox, and the ABox are all fixed, and just the RBox varies.

Theorem 30. *For $\mathcal{K}$ a regular $\mathcal{ELRO}^+$ knowledge base and q a Boolean conjunctive query, checking $\mathcal{K} \models q$ is* PSpace*-hard even when*

- *the query is fixed and consist of two binary atoms over a single quantified variable,*
- *the TBox is fixed and contains only axioms of the form $A \sqsubseteq \exists S.A$, and*
- *the ABox is fixed and contains a single unary assertion.*

Proof. We reduce the PSpace-hard problem of deciding whether the intersection of the languages generated by m deterministic finite automata is nonempty (Kozen, 1977). Let $\mathcal{F}'_1, \dots, \mathcal{F}'_m$ be deterministic finite automata over alphabet Σ', let σ_1 and σ_2 be fresh symbols not occurring in Σ', and let $\Sigma = \Sigma' \cup \{\sigma_1, \sigma_2\}$. For each $j \in [1..m]$, let $\mathcal{F}_j = \langle Q_j, \Sigma, \delta_j, i_j, f_j \rangle$ be the deterministic finite automaton over alphabet Σ obtained by extending $\mathcal{F}'_j$ with a transition labelled by σ_1 from the final state f'_j of $\mathcal{F}'_j$ to itself, and with a transition labelled by σ_2 from f'_j to a fresh final state f_j of $\mathcal{F}_j$. Then, $\bigcap_j \mathcal{L}(\mathcal{F}'_j) \neq \emptyset$ if and only if a word $w \in \bigcap_j \mathcal{L}(\mathcal{F}_j)$ exist such that $|w|$ is odd: given $w \in \bigcap_j \mathcal{L}(\mathcal{F}'_j)$, if $|w|$ is odd, then $|w \cdot \sigma_1 \cdot \sigma_2|$ is odd and $w \cdot \sigma_1 \cdot \sigma_2 \in \mathcal{L}(\mathcal{F}_j)$ for each $j \in [1..m]$, and if $|w|$ is even, then $|w \cdot \sigma_2|$ is odd and $w \cdot \sigma_2 \in \mathcal{L}(\mathcal{F}_j)$ for each $j \in [1..m]$. Finally, we assume w.l.o.g. that $Q_i \cap Q_j \neq \emptyset$ and $Q_i \subseteq \mathsf{R}$ hold for each $1 \leq i < j \leq m$, and that $\Sigma \subseteq \mathsf{R}$ as well.

Let $w = S_1 \cdots S_n$ be a word in Σ^* such that n is odd, and let $\Gamma = \Sigma \cup Q_1 \cup \ldots \cup Q_m$. Clearly, $w \in \bigcap_j \mathcal{L}(\mathcal{F}_j)$ holds if and only if a word $\rho_w \in \Gamma^*$ of the form

$$\rho_w = e_1^0 \cdots e_m^0 \cdot S_1 \cdot o_m^1 \cdots o_1^1 \cdot S_2 \cdot e_1^2 \cdots e_m^2 \cdots\cdots e_1^{n-1} \cdots e_m^{n-1} \cdot S_n \cdot o_m^n \cdots o_1^n \qquad (46)$$

exists such that the following conditions hold for each $j \in [1..m]$:

(i) for each $i \in [1..n]$ with i odd, we have $o_j^i \in Q_j$ and $\delta_j(e_j^{i-1}, S_i) = o_j^i$;

(ii) for each $i \in [1..n]$ with i even, we have $e_j^i \in Q_j$ and $\delta_j(o_j^{i-1}, S_i) = e_j^i$; and

(iii) $e_j^0 = i_j$ and $o_j^n = f_j$.

Now let $\mathcal{L}_O$, $\mathcal{L}_E$, $\mathcal{L}_1$, and $\mathcal{L}_2$ be the following languages.

$$\mathcal{L}_O := \{e_1 \cdots e_m \cdot S \cdot o_m \cdots o_1 \mid S \in \Sigma \text{ and } \delta_j(e_j, S) = o_j, \text{ for each } j \in [1..m]\} \tag{47}$$
$$\mathcal{L}_E := \{o_m \cdots o_1 \cdot S \cdot e_1 \cdots e_m \mid S \in \Sigma \text{ and } \delta_j(o_j, S) = e_j, \text{ for each } j \in [1..m]\} \tag{48}$$
$$\mathcal{L}_1 := (\mathcal{L}_O \cdot \Sigma)^* \cdot \mathcal{L}_O \tag{49}$$
$$\mathcal{L}_2 := \{i_1 \cdots i_m\} \cdot (\Sigma \cdot \mathcal{L}_E)^* \cdot \Sigma \cdot \{f_m \cdots f_1\} \tag{50}$$

Consider an arbitrary word $w \in \Sigma^*$ and the corresponding word $\rho_w \in \Gamma^*$. By the definition of $\mathcal{L}_1$, we have that $\rho_w \in \mathcal{L}_1$ if ρ_w is of the form (46) and it satisfies property (i). Similarly, by the definition of $\mathcal{L}_2$, we have that $\rho_w \in \mathcal{L}_2$ if ρ_w is of the form (46) and it satisfies properties (ii) and (iii). Thus, $\rho_w \in \mathcal{L}_1 \cap \mathcal{L}_2$ if and only if $w \in \bigcap_j \mathcal{L}(\mathcal{F}_j)$. For simplicity, in the rest of this proof, we will use the following equivalent formulations of $\mathcal{L}_1$ and $\mathcal{L}_2$.

$$\mathcal{L}_1 = \mathcal{L}_O \cup (\mathcal{L}_O \cdot \Sigma)^+ \cdot \mathcal{L}_O \tag{51}$$
$$\mathcal{L}_2 = \{i_1 \cdots i_m\} \cdot \Sigma \cdot \{f_m \cdots f_1\} \cup \{i_1 \cdots i_m\} \cdot (\Sigma \cdot \mathcal{L}_E)^+ \cdot \Sigma \cdot \{f_m \cdots f_1\} \tag{52}$$

We next define a knowledge base $\mathcal{K}$ and a fixed query q such that $\mathcal{K} \models q$ if and only if $\bigcap_j \mathcal{L}(\mathcal{F}_j) \neq \emptyset$. We will present our construction in stages, and for each we will describe how it affects the canonical model $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$ of $\mathcal{K}$—that is, the model constructed using the standard notion of chase (i.e., as in Definition 25, but with all semantic conditions on $\mathcal{K}$ replaced by the syntactic checks for axioms in $\mathcal{K}$). For simplicity, we first present $\mathcal{K}$ in which the TBox depends on Γ, and later we modify the encoding to use a fixed TBox.

The TBox $\mathcal{T}$ contains axioms (53), and the ABox $\mathcal{A}$ contains only axiom (54). We assume that $a_\epsilon^\mathcal{I} = a_\epsilon$; then, for each word $\rho \in \Gamma^*$, a domain element a_ρ exists that is connected to a_ϵ via a chain of roles corresponding to ρ.

$$A \sqsubseteq \exists \omega . A \qquad \text{for each symbol } \omega \in \Gamma \tag{53}$$
$$A(a_\epsilon) \tag{54}$$

We next present an RBox $\mathcal{R}$ consisting of four parts, each encoding languages $\mathcal{L}_O$, $\mathcal{L}_E$, $\mathcal{L}_1$, and $\mathcal{L}_2$. Our encoding uses fresh roles $L_O^{S,1}, \ldots, L_O^{S,m+1}$ and $L_E^{S,0}, \ldots, L_E^{S,m}$ uniquely associated with each role $S \in \Sigma$, as well as fresh roles L_O, L_E, L_Σ, L_1, L_1', L_2, and L_2'.

The first part of $\mathcal{R}$ contains axioms (55)–(57). It should be clear that, for all words $\rho_1, \rho_2 \in \Gamma^*$ where ρ_1 is a prefix of ρ_2, we have $\langle a_{\rho_1}, a_{\rho_2} \rangle \in L_O^\mathcal{I}$ if and only if $\rho_2 - \rho_1 \in \mathcal{L}_O$.

$$S \sqsubseteq L_O^{S,m+1} \qquad \forall S \in \Sigma \tag{55}$$
$$e_j \cdot L_O^{S,j+1} \cdot o_j \sqsubseteq L_O^{S,j} \qquad \forall j \in [1..m]\ \forall S \in \Sigma\ \forall e_j, o_j \in Q_j \text{ with } \delta_j(e_j, S) = o_j \tag{56}$$
$$L_O^{S,1} \sqsubseteq L_O \tag{57}$$

The second part of $\mathcal{R}$ contains axioms (58)–(60). It should be clear that, for all words $\rho_1, \rho_2 \in \Gamma^*$ where ρ_1 is a prefix of ρ_2, we have $\langle a_{\rho_1}, a_{\rho_2} \rangle \in L_E^{\mathcal{I}}$ if and only if $\rho_2 - \rho_1 \in \mathcal{L}_E$.

$$S \sqsubseteq L_E^{S,0} \qquad \forall S \in \Sigma \tag{58}$$

$$o_j \cdot L_E^{S,j-1} \cdot e_j \sqsubseteq L_E^{S,j} \qquad \forall j \in [1..m]\ \forall S \in \Sigma\ \forall e_j, o_j \in Q_j \text{ with } \delta_j(o_j, S) = e_j \tag{59}$$

$$L_E^{S,m} \sqsubseteq L_E \tag{60}$$

The third part of $\mathcal{R}$ contains axioms (61)–(65). It should be clear that, for all words $\rho_1, \rho_2 \in \Gamma^*$ where ρ_1 is a prefix of ρ_2, we have $\langle a_{\rho_1}, a_{\rho_2} \rangle \in L_1^{\mathcal{I}}$ if and only if $\rho_2 - \rho_1 \in \mathcal{L}_1$.

$$S \sqsubseteq L_\Sigma \qquad \forall S \in \Sigma \tag{61}$$

$$L_O \sqsubseteq L_1 \tag{62}$$

$$L_O \cdot L_\Sigma \sqsubseteq L_1' \tag{63}$$

$$L_1' \cdot L_1' \sqsubseteq L_1' \tag{64}$$

$$L_1' \cdot L_O \sqsubseteq L_1 \tag{65}$$

The fourth part of $\mathcal{R}$ contains axioms (66)–(69). It should be clear that, for all words $\rho_1, \rho_2 \in \Gamma^*$ where ρ_1 is a prefix of ρ_2, we have $\langle a_{\rho_1}, a_{\rho_2} \rangle \in L_2^{\mathcal{I}}$ if and only if $\rho_2 - \rho_1 \in \mathcal{L}_2$.

$$i_1 \cdots i_m \cdot L_\Sigma \cdot f_m \cdots f_1 \sqsubseteq L_2 \tag{66}$$

$$L_\Sigma \cdot L_E \sqsubseteq L_2' \tag{67}$$

$$L_2' \cdot L_2' \sqsubseteq L_2' \tag{68}$$

$$i_1 \cdots i_m \cdot L_2' \cdot L_\Sigma \cdot f_m \cdots f_1 \sqsubseteq L_2 \tag{69}$$

Query q is given in (70). Then, $\mathcal{K} \models q$ if and only if a word $\rho \in \Gamma^*$ exists such that $\langle a_\epsilon, a_\rho \rangle \in L_1^{\mathcal{I}}$ and $\langle a_\epsilon, a_\rho \rangle \in L_2^{\mathcal{I}}$, and the latter is clearly the case if and only if $\rho \in \mathcal{L}_1 \cap \mathcal{L}_2$. RBox $\mathcal{R}$ is regular and of size polynomial in the size of automata $\mathcal{F}_1, \ldots, \mathcal{F}_m$.

$$q = \exists y.\, L_1(a_\epsilon, y) \wedge L_2(a_\epsilon, y) \tag{70}$$

We next tighten this reduction to use the fixed TBox $\mathcal{T}'$ consisting of axioms (71)–(72), where P_0 and P_1 are fresh roles.

$$A \sqsubseteq \exists P_0.A \tag{71}$$

$$A \sqsubseteq \exists P_1.A \tag{72}$$

Now let $k = \lceil \log_2 |\Gamma| \rceil$, and assume that each symbol $\omega \in \Gamma$ corresponds to a k-digit binary number $b_1 \cdots b_k$ with $b_i \in \{0, 1\}$. Then, let $\mathcal{R}'$ be $\mathcal{R}$ extended with axioms (73).

$$P_{b_1} \cdots P_{b_k} \sqsubseteq \omega \qquad \text{for each } \omega \in \Gamma \text{ corresponding to } b_1 \cdots b_k \tag{73}$$

Finally, let $\mathcal{K}' = \langle \mathcal{T}', \mathcal{R}', \mathcal{A} \rangle$, and let $\mathcal{I}'$ be the canonical model of $\mathcal{K}'$. Axioms (54), (71), and (72) ensure existence of a binary tree whose edges are labelled with roles P_0 and P_1. Furthermore, axioms (73) ensure that, for each $\omega \in \Gamma$ and each sequence of k edges in this tree corresponding to the binary number assigned to ω, there is a 'shortcut' in the tree labelled with ω. Thus, $\mathcal{I}$ can be homomorphically embedded into $\mathcal{I}'$. Finally, roles P_0 and P_1 do not occur in $\mathcal{R}$ and query q checks for existence of a domain element connected to a_ϵ; therefore, the 'extra' edges in $\mathcal{I}'$ are irrelevant. Consequently, the encoding of languages $\mathcal{L}_1$ and $\mathcal{L}_2$ works in the same way as with the varying TBox $\mathcal{T}$. $\square$

Finally, we characterise the complexity of BCQ answering over $\mathcal{ELRO}^+$ knowledge bases.

Theorem 31. *For $\mathcal{K}$ a regular $\mathcal{ELRO}^+$ KB and q a Boolean CQ, checking $\mathcal{K} \models q$ is*

1. PTime-*complete in data complexity,*
2. NP-*complete, if the RBox $\mathcal{R}$ is fixed, and*
3. PSpace-*complete in combined complexity.*

Proof. Calvanese et al. (2006) proved that BCQ answering is PTime-hard in data complexity already for $\mathcal{EL}$ knowledge bases. Furthermore, when the query is not fixed, BCQ answering is NP-hard already over relational databases (Chandra & Merlin, 1977). Then the theorem follows by Theorems 23 and 30, and by Savitch's theorem. $\square$

6. Navigational Queries

The data in DL knowledge bases has graph-like structure, where unary assertions encode properties of graph nodes and binary assertions encode graph edges. Conjunctive queries cannot express recursive properties such as reachability, and so their expressivity is often insufficient in applications that require graph navigation. As the popularity of graph databases is on the rise, a number of navigational languages for querying graph-like data have been proposed; for example, regular path queries (Barceló, 2013) use regular expressions to express complex navigational patterns between graph vertices, and graph XPath queries (Libkin et al., 2013) extend regular path queries with the converse operator, negation on regular expressions, and checking properties of vertices using Boolean combinations of concepts and existential quantifications over paths. In the DL context, the computational complexity of navigational queries has been studied for several expressive DLs and members of the DL-Lite family and the $\mathcal{EL}(\mathcal{H})$ fragment of $\mathcal{ELRO}^+$ (Calvanese, Eiter, & Ortiz, 2009; Bienvenu et al., 2013; Kostylev et al., 2014; Bienvenu et al., 2014). In order to complete the complexity landscape of this problem, in this section we study the problem of answering graph XPath queries over $\mathcal{ELRO}^+$ knowledge bases.

6.1 Graph XPath Queries

Graph XPath queries consist of *node expressions* and *path expressions*, whose syntaxes are defined respectively by the following two context-free grammars for B a basic concept and S a role.

$$\begin{aligned}
&\varphi \to B \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \langle\alpha\rangle \\
&\alpha \to S \mid S^- \mid \alpha_1 \cdot \alpha_2 \mid \alpha_1 + \alpha_2 \mid \alpha^* \mid \overline{\alpha} \mid \mathsf{test}(\varphi)
\end{aligned}$$

Following Libkin et al. (2013), we consider the following expression fragments.

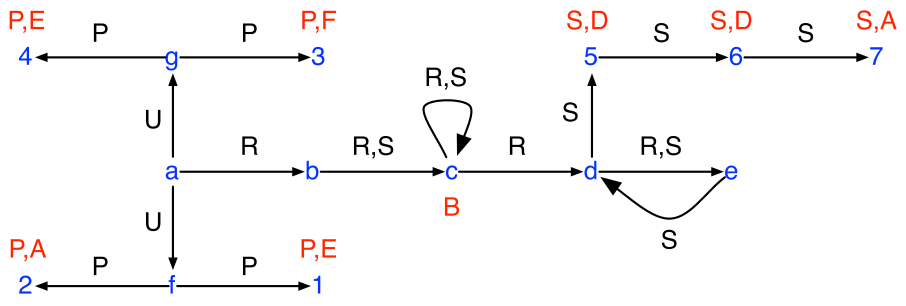

Figure 8: Interpretation $\mathcal{I}$

1. The *path-positive* fragment disallows path expressions of the form $\overline{\alpha}$.
2. The *positive* fragment disallows path expressions of the form $\overline{\alpha}$ and node expressions of the form $\neg\varphi$.
3. The *converse-free* fragment disallows path expressions of the form S^-.

A *graph XPath atom* has the form $\varphi(s)$ or $\alpha(s,t)$, for φ a node expression, α a path expression, and s and t terms. A *conjunctive graph XPath query* (CGXQ) g is an expression $g = \exists\vec{y}.\ \psi(\vec{x},\vec{y})$ where ψ is a conjunction of graph atoms over variables $\vec{x} \cup \vec{y}$; variables $\vec{x}$ are called the *answer variables* of g. If $\vec{x} = \emptyset$, then $g = \exists\vec{y}.\ \psi(\vec{y})$ is a Boolean CGXQ. Path-positive, positive, and converse-free CGXQs are obtained by restricting query atoms accordingly. Finally, a *graph XPath query* (GXQ) is a CGXQ containing a single atom.

To define the semantics of CGXQs, let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ be a first-order interpretation. The interpretation of node and path expressions in $\mathcal{I}$ is inductively defined as follows.

$$\begin{aligned}
(\neg\varphi)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus (\varphi)^{\mathcal{I}} \\
(\varphi_1 \wedge \varphi_2)^{\mathcal{I}} &= (\varphi_1)^{\mathcal{I}} \cap (\varphi_2)^{\mathcal{I}} \\
(\varphi_1 \vee \varphi_2)^{\mathcal{I}} &= (\varphi_1)^{\mathcal{I}} \cup (\varphi_2)^{\mathcal{I}} \\
(\langle\alpha\rangle)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} : \langle x,y\rangle \in \alpha^{\mathcal{I}}\}
\end{aligned}
\qquad
\begin{aligned}
(S^-)^{\mathcal{I}} &= \{\langle y,x\rangle \mid \langle x,y\rangle \in S^{\mathcal{I}}\} \\
(\alpha_1 \cdot \alpha_2)^{\mathcal{I}} &= (\alpha_1)^{\mathcal{I}} \circ (\alpha_2)^{\mathcal{I}} \\
(\alpha_1 + \alpha_2)^{\mathcal{I}} &= (\alpha_1)^{\mathcal{I}} \cup (\alpha_2)^{\mathcal{I}} \\
(\alpha^*)^{\mathcal{I}} &= (\alpha^{\mathcal{I}})^* \\
(\overline{\alpha})^{\mathcal{I}} &= \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \setminus (\alpha)^{\mathcal{I}} \\
(\mathsf{test}(\varphi))^{\mathcal{I}} &= \{\langle x,x\rangle \mid x \in \varphi^{\mathcal{I}}\}
\end{aligned}$$

Please observe that the difference of path expressions α_1 and α_2 corresponds to $\overline{(\overline{\alpha_1} + \alpha_2)}$, whereas the intersection of α_1 and α_2 corresponds to $\overline{(\overline{\alpha_1} + \overline{\alpha_2})}$; moreover, Libkin et al. (2013) define a path expression ϵ, which in our setting corresponds to $\mathsf{test}(\top_c)$. Satisfaction of a Boolean CGXQ g in $\mathcal{I}$ and CGXQ entailment are defined in the obvious way; moreover, *Boolean CGXQ answering* is the problem of checking $\mathcal{K} \models g$.

Example 32. *We illustrate these definitions using interpretation $\mathcal{I}$ shown in Figure 8; notation is as in Example 15. Moreover, let α_1, α_2, and α_3 be the following path expressions.*

$$\alpha_1 = (R \cdot \mathsf{test}(\langle S^* \cdot \mathsf{test}(A \vee B)\rangle))^* \tag{74}$$
$$\alpha_2 = (U \cdot \mathsf{test}(\neg\langle P \cdot \mathsf{test}(A \vee B)\rangle)) \tag{75}$$

Node expressions	
$\mathcal{T}_B = \{B \sqsubseteq C_B\}$	$\mathcal{R}_B = \emptyset$
$\mathcal{T}_{\varphi_1 \wedge \varphi_2} = \{C_{\varphi_1} \sqcap C_{\varphi_1} \sqsubseteq C_{\varphi_1 \wedge \varphi_2}\} \cup \mathcal{T}_{\varphi_1} \cup \mathcal{T}_{\varphi_2}$	$\mathcal{R}_{\varphi_1 \wedge \varphi_2} = \mathcal{R}_{\varphi_1} \cup \mathcal{R}_{\varphi_2}$
$\mathcal{T}_{\varphi_1 \vee \varphi_2} = \{C_{\varphi_1} \sqsubseteq C_{\varphi_1 \vee \varphi_2},\ C_{\varphi_2} \sqsubseteq C_{\varphi_1 \vee \varphi_2}\} \cup \mathcal{T}_{\varphi_1} \cup \mathcal{T}_{\varphi_2}$	$\mathcal{R}_{\varphi_1 \vee \varphi_2} = \mathcal{R}_{\varphi_1} \cup \mathcal{R}_{\varphi_2}$
$\mathcal{T}_{\langle \alpha \rangle} = \{\exists T_\alpha.\top_c \sqsubseteq C_{\langle \alpha \rangle}\} \cup \mathcal{T}_\alpha$	$\mathcal{R}_{\langle \alpha \rangle} = \mathcal{R}_\alpha$

Path expressions	
$\mathcal{T}_S = \emptyset$	$\mathcal{R}_S = \{S \sqsubseteq T_S\}$
$\mathcal{T}_{\alpha_1 \cdot \alpha_2} = \mathcal{T}_{\alpha_1} \cup \mathcal{T}_{\alpha_2}$	$\mathcal{R}_{\alpha_1 \cdot \alpha_2} = \{T_{\alpha_1} \cdot T_{\alpha_2} \sqsubseteq T_{\alpha_1 \cdot \alpha_2}\} \cup \mathcal{R}_{\alpha_1} \cup \mathcal{R}_{\alpha_2}$
$\mathcal{T}_{\alpha_1 + \alpha_2} = \mathcal{T}_{\alpha_1} \cup \mathcal{T}_{\alpha_2}$	$\mathcal{R}_{\alpha_1 + \alpha_2} = \{T_{\alpha_1} \sqsubseteq T_{\alpha_1 + \alpha_2},\ T_{\alpha_2} \sqsubseteq T_{\alpha_1 + \alpha_2}\} \cup \mathcal{R}_{\alpha_1} \cup \mathcal{R}_{\alpha_2}$
$\mathcal{T}_{\alpha^*} = \mathcal{T}_\alpha$	$\mathcal{R}_{\alpha^*} = \{\epsilon \sqsubseteq T_{\alpha^*},\ T_\alpha \sqsubseteq T_{\alpha^*},\ T_{\alpha^*} \cdot T_{\alpha^*} \sqsubseteq T_{\alpha^*}\} \cup \mathcal{R}_\alpha$
$\mathcal{T}_{\mathsf{test}(\varphi)} = \{C_\varphi \sqsubseteq \exists T_{\mathsf{test}(\varphi)}.\mathsf{Self}\} \cup \mathcal{T}_\varphi$	$\mathcal{R}_{\mathsf{test}(\varphi)} = \mathcal{R}_\varphi$

Table 6: Encoding positive, converse-free node and path expressions using axioms

$$\alpha_3 = (\overline{(R \cdot S)^*}) \tag{76}$$

Expression α_1 is positive, and it retrieves all pairs of individuals that are connected by a path of R-edges such that, for each element occurring in the path other than the first, there exists an outgoing path of S-edges reaching a member of concept $A \sqcup B$. For example, we have $\{\langle a^\mathcal{I}, d^\mathcal{I}\rangle, \langle a^\mathcal{I}, e^\mathcal{I}\rangle\} \subseteq (\alpha_1)^\mathcal{I}$.

In contrast, expression α_2 is path-positive, and it retrieves all pairs of individuals that are connected by a U-edge such that no P-successor exists that is a member of concept $A \sqcup B$. For example, we have $\langle a^\mathcal{I}, g^\mathcal{I}\rangle \in (\alpha_2)^\mathcal{I}$, but $\langle a^\mathcal{I}, f^\mathcal{I}\rangle \notin (\alpha_2)^\mathcal{I}$.

Finally, expression α_3 is neither positive nor path-positive, and it retrieves all pairs of individuals that are connected by a path not *consisting of a sequence of edges described by the regular expression $(R \cdot S)^*$. For example, we have $\langle a^\mathcal{I}, d^\mathcal{I}\rangle \in (\alpha_3)^\mathcal{I}$, but $\langle a^\mathcal{I}, e^\mathcal{I}\rangle \notin (\alpha_3)^\mathcal{I}$.*

Let $g = \exists x, y, z.\alpha_1(x, y) \wedge \alpha_2(x, z) \wedge \alpha_3(x, y)$ be a conjunctive graph XPath query, and let $\sigma = \{x \mapsto a, y \mapsto d, z \mapsto g\}$ be a substitution. Using Figure 8, one can check that $\mathcal{I} \models \sigma(g)$.

As observed by Kostylev et al. (2014), node expressions in graph XPath queries correspond precisely to formulas in propositional dynamic logic with negation (PDL$^\neg$) (Harel et al., 2000); the satisfiability problem for PDL$^\neg$ is undecidable (Harel, 1984), so answering GXQs under DL constraints is undecidable. Decidability results have been recently obtained for path-positive and positive queries over *DL-Lite* knowledge bases (Kostylev et al., 2014). In addition, Kostylev et al. (2014) proved that, for all DLs, answering path-positive, converse-free GXQs is coNP-hard in data-complexity. Finally, Bienvenu et al. (2014) proved that answering positive GXQs over $\mathcal{EL}$ knowledge bases is ExpTime-complete. Thus, in the rest of this section we focus on positive, converse-free graph XPath queries.

6.2 Complexity of Answering Positive, Converse-Free Graph XPath Queries

In the rest of this section, we fix an $\mathcal{ELRO}^+$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$ such that $\mathcal{R}$ is regular. We next show that, given a positive, converse-free Boolean CGXQ g, one can construct in polynomial time a regular $\mathcal{ELRO}^+$ KB $\mathcal{K}'$ and a Boolean CQ q' such that $\mathcal{K} \models g$ if and only if $\mathcal{K}' \models q'$. Our construction of $\mathcal{K}'$ combines various expressive features of $\mathcal{ELRO}^+$:

role inclusions and reflexive roles encode the path expressions of g in the RBox, and self-restrictions encode the node expressions of g in the TBox.

Proposition 33. *Given a positive, converse-free Boolean CGXQ g over $\mathcal{K}$, one can compute in time polynomial in $|\mathcal{K}|+|g|$ an $\mathcal{ELRO}^+$ KB $\mathcal{K}'$ and a Boolean CQ q' such that the RBox of $\mathcal{K}'$ is regular, g and q' have equally many atoms, and $\mathcal{K} \models g$ if and only if $\mathcal{K}' \models q'$.*

Proof. Let $g = \exists \vec{y}.\ \psi(\vec{y})$ be a positive, converse-free Boolean CGXQ over $\mathcal{K}$. For each positive node expression φ, let C_φ be a fresh atomic concept uniquely associated with φ and, for each positive, converse-free path expression α, let T_α be a fresh role uniquely associated with α. By structural induction, we associate with each φ (resp. α) a TBox $\mathcal{T}_\varphi$ and an RBox $\mathcal{R}_\varphi$ (resp. a TBox $\mathcal{T}_\alpha$ and an RBox $\mathcal{R}_\alpha$) as shown in Table 6. Then, let $\mathcal{K}' = \langle \mathcal{T} \cup \mathcal{T}', \mathcal{R} \cup \mathcal{R}', \mathcal{A} \rangle$ where TBox $\mathcal{T}'$ and RBox $\mathcal{R}'$ are as follows.

$$\mathcal{T}' = \bigcup_{\varphi(s)\in\psi} \mathcal{T}_\varphi \cup \bigcup_{\alpha(s,t)\in\psi} \mathcal{T}_\alpha \qquad\qquad \mathcal{R}' = \bigcup_{\varphi(s)\in\psi} \mathcal{R}_\varphi \cup \bigcup_{\alpha(s,t)\in\psi} \mathcal{R}_\alpha$$

Now let $q' = \exists \vec{y}.\ \psi'(\vec{y})$ be the Boolean CQ where ψ' contains $C_\varphi(s)$ for each atom $\varphi(s) \in \psi$ and $T_\alpha(s,t)$ for each atom $\alpha(s,t) \in \psi$. Clearly, g and q' have the same number of atoms; moreover, since query g is over $\mathcal{K}$, query q' is over $\mathcal{K}'$. Finally, both q' and $\mathcal{K}'$ can be computed in polynomial time in the input size, and the RBox of $\mathcal{K}'$ is clearly regular. We next show that $\mathcal{K}' \not\models q'$ if and only if $\mathcal{K} \not\models g$.

($\Rightarrow$) Assume that $\mathcal{K}' \not\models q'$, so an interpretation $\mathcal{I}$ exists such that $\mathcal{I} \models \mathcal{K}'$ and $\mathcal{I} \not\models q'$. Since each axiom of $\mathcal{K}$ is also an axiom of $\mathcal{K}'$, we have that $\mathcal{I} \models \mathcal{K}$. Furthermore, for each positive node expression φ and positive path expression α, we have $\varphi^\mathcal{I} \subseteq (C_\varphi)^\mathcal{I}$ and $\alpha^\mathcal{I} \subseteq (T_\alpha)^\mathcal{I}$. We prove this claim by simultaneous induction on the structure of node and path expressions.

Base case. For the base case, let φ be an arbitrary node expression of the form $\varphi = B$ and let α be an arbitrary path expression of the form $\alpha = S$. Since $B \sqsubseteq C_B \in \mathcal{T}'$, $S \sqsubseteq T_S \in \mathcal{R}'$, and $\mathcal{I}$ is a model of $\mathcal{K}'$, the claim easily follows.

Inductive step. For the inductive step, we distinguish two cases.

First, consider an arbitrary node expression φ such that the property holds for all node and path expressions occurring in φ. Then let x be an arbitrary element of $\Delta^\mathcal{I}$ and assume that $x \in \varphi^\mathcal{I}$; we show that $x \in C_\varphi^\mathcal{I}$ by considering the various forms that φ can take.

- $\varphi = \varphi_1 \wedge \varphi_2$. Since $x \in \varphi^\mathcal{I}$, we have $x \in \varphi_1^\mathcal{I}$ and $x \in \varphi_2^\mathcal{I}$. By the inductive hypothesis, we have $x \in C_{\varphi_1}^\mathcal{I}$ and $x \in C_{\varphi_2}^\mathcal{I}$. By the definition of $\mathcal{T}'$, we have $C_{\varphi_1} \sqcap C_{\varphi_2} \sqsubseteq C_\varphi \in \mathcal{T}'$. Since $\mathcal{I}$ is a model of $\mathcal{T}'$, we have $x \in C_\varphi^\mathcal{I}$, as required.

- $\varphi = \varphi_1 \vee \varphi_2$. The proof for this case is similar to the one above.

- $\varphi = \langle \alpha \rangle$. Since $x \in \varphi^\mathcal{I}$, there exists $y \in \Delta^\mathcal{I}$ such that $\langle x, y \rangle \in \alpha^\mathcal{I}$. By the inductive hypothesis, we have $\langle x, y \rangle \in T_\alpha^\mathcal{I}$. By the definition of $\mathcal{T}'$, we have $\exists T_\alpha.\top_c \sqsubseteq C_\varphi \in \mathcal{T}'$. Since $\mathcal{I}$ is a model of $\mathcal{T}'$, we have $x \in C_\varphi^\mathcal{I}$, as required.

Second, consider an arbitrary path expression α such that the property holds for all node and path expressions occurring in α. Then let x and y be arbitrary elements of $\Delta^\mathcal{I}$ and assume that $\langle x, y \rangle \in \alpha^\mathcal{I}$; we show that $\langle x, y \rangle \in T_\alpha^\mathcal{I}$ by considering the various forms that α can take.

- $\alpha = \alpha_1 \cdot \alpha_2$. Since $\langle x, y\rangle \in \alpha^{\mathcal{I}}$, there exists $z \in \Delta^{\mathcal{I}}$ such that $\langle x, z\rangle \in \alpha_1^{\mathcal{I}}$ and $\langle z, y\rangle \in \alpha_2^{\mathcal{I}}$. By the inductive hypothesis, we have $\langle x, z\rangle \in T_{\alpha_1}^{\mathcal{I}}$ and $\langle z, y\rangle \in T_{\alpha_2}^{\mathcal{I}}$. Moreover, by the definition of $\mathcal{R}'$, we have $T_{\alpha_1} \cdot T_{\alpha_2} \sqsubseteq T_\alpha \in \mathcal{R}'$. Since $\mathcal{I}$ is a model of $\mathcal{R}'$, we have $\langle x, y\rangle \in T_\alpha^{\mathcal{I}}$, as required.

- $\alpha = \alpha_1 + \alpha_2$. Since $\langle x, y\rangle \in \alpha^{\mathcal{I}}$, we have that $\langle x, y\rangle \in \alpha_1^{\mathcal{I}}$ or $\langle x, y\rangle \in \alpha_2^{\mathcal{I}}$. By the inductive hypothesis, we have $\langle x, y\rangle \in T_{\alpha_1}^{\mathcal{I}}$ or $\langle x, y\rangle \in T_{\alpha_2}^{\mathcal{I}}$. By the definition of $\mathcal{R}'$, we have $\{T_{\alpha_1} \sqsubseteq T_\alpha, T_{\alpha_2} \sqsubseteq T_\alpha\} \subseteq \mathcal{R}'$. Since $\mathcal{I}$ is a model of $\mathcal{R}'$, we have $\langle x, y\rangle \in T_\alpha^{\mathcal{I}}$.

- $\alpha = \alpha_1^*$. First, consider the case in which $x = y$. By the definition of $\mathcal{R}'$, we have $\epsilon \sqsubseteq T_\alpha \in \mathcal{R}'$. Since $\mathcal{I}$ is a model of $\mathcal{R}'$, we have $\langle x, y\rangle \in T_\alpha^{\mathcal{I}}$, as required. Otherwise, consider the case in which $x \neq y$. Since $\langle x, y\rangle \in \alpha^{\mathcal{I}}$, elements $x_0, \ldots, x_n$ with $x_0 = x$ and $x_n = y$ exist in $\Delta^{\mathcal{I}}$ such that $n > 0$ and $\langle x_{i-1}, x_i\rangle \in \alpha_1^{\mathcal{I}}$ for each $i \in [1..n]$. By the inductive hypothesis, for each $i \in [1..n]$, we have $\langle x_{i-1}, x_i\rangle \in T_{\alpha_1}^{\mathcal{I}}$. By the definition of $\mathcal{R}'$, we have $T_{\alpha_1} \cdot T_{\alpha_1} \sqsubseteq T_\alpha \in \mathcal{R}'$. Since $\mathcal{I}$ is a model of $\mathcal{R}'$, we have $\langle x, y\rangle \in T_\alpha$.

- $\alpha = \mathsf{test}(\varphi)$. It follows that $x = y$ and that $x \in \varphi^{\mathcal{I}}$. By the inductive hypothesis, we have $x \in C_\varphi^{\mathcal{I}}$. By the definition of $\mathcal{T}'$, we have $C_\varphi \sqsubseteq \exists T_\alpha.\mathsf{Self} \in \mathcal{T}'$. Since $\mathcal{I}$ is a model of $\mathcal{T}'$, we have $\langle x, y\rangle \in T_\alpha^{\mathcal{I}}$, as required.

But then, since node and path expressions in g are positive, $\mathcal{I} \not\models q'$ implies $\mathcal{I} \not\models g$.

($\Leftarrow$) Assume that $\mathcal{K} \not\models g$, so an interpretation $\mathcal{I}$ exists such that $\mathcal{I} \models \mathcal{K}$ and $\mathcal{I} \not\models g$. Let $\mathcal{I}'$ be the interpretation obtained by extending $\mathcal{I}$ to the fresh concepts and roles as follows.

$$(C_\varphi)^{\mathcal{I}'} = \varphi^{\mathcal{I}'} \qquad\qquad (T_\alpha)^{\mathcal{I}'} = \alpha^{\mathcal{I}'}$$

By the definition of $\mathcal{K}'$, it is straightforward to see that $\mathcal{I}' \models \mathcal{K}'$; furthermore, by the definition of q', it is straightforward to see that $\mathcal{I}' \not\models q'$, as required. $\square$

Next, we establish the complexity of answering positive, converse-free (C)GXQs over $\mathcal{ELRO}^+$ knowledge bases.

Theorem 34. *For $\mathcal{K}$ a regular $\mathcal{ELRO}^+$ KB and g a positive, converse-free Boolean CGXQ, checking $\mathcal{K} \models g$ is* PTime*-complete in data complexity, and* PSpace*-complete in combined complexity. For g a positive, converse-free Boolean GXQ, checking $\mathcal{K} \models g$ is* PTime*-complete in combined and data complexities.*

Proof. The hardness in data complexity of Boolean positive, converse-free (C)GXQs follows from the PTime-hardness of instance checking in $\mathcal{EL}$ (Calvanese et al., 2006).

For positive, converse-free GXQs, hardness in combined complexity is inherited from the PTime-hardness of TBox reasoning in $\mathcal{EL}$ (Baader et al., 2005). For the matching upper bounds, Proposition 33 allows us to reduce Boolean GXQ answering to checking entailments of the form $\mathcal{K}' \models q'$ where q' is a BCQ containing only one atom. We next show that, for each possible form of q', we can reduce the latter problem to checking entailment of $\mathcal{ELRO}^+$ concept inclusions, which can be decided in PTime. In the following, c is an arbitrarily chosen individual from $\mathsf{I}_{\mathcal{K}'}$.

- $\mathcal{K}' \models A(a)$ if and only if $\mathcal{K}' \models \{a\} \sqsubseteq A$.

- $\mathcal{K}' \models \exists y.A(y)$ if and only if $\mathcal{K}' \models \{c\} \sqsubseteq \exists \top_r.A$.
- $\mathcal{K}' \models S(a,b)$ if and only if $\mathcal{K}' \models \{a\} \sqsubseteq \exists S.\{b\}$.
- $\mathcal{K}' \models \exists y.S(y,b)$ if and only if $\mathcal{K}' \models \{c\} \sqsubseteq \exists \top_r.\exists S.\{b\}$.
- $\mathcal{K}' \models \exists y.S(a,y)$ if and only if $\mathcal{K}' \models \{a\} \sqsubseteq \exists S.\top_c$.
- $\mathcal{K}' \models \exists y_1, y_2.S(y_1,y_2)$ if and only if $\mathcal{K}' \models \{c\} \sqsubseteq \exists \top_r.\exists S.\top_c$.

For positive, converse-free CGXQs, hardness in combined complexity is given by Theorem 31, and the matching upper bounds follow from Theorem 23 and Proposition 33. □

7. Conclusions

In this paper, we presented the first CQ answering algorithm for OWL 2 EL that runs in PSpace, thus closing a longstanding open question. Our algorithm is based on an innovative, succinct encoding of regular role inclusions using bounded-stack PDA—that is, finite automata extended with a stack of fixed size. We believe this encoding is interesting in its own right, as it can be used to optimise popular OWL 2 DL reasoners. Moreover, we refined the previously known PSpace lower bound for CQ answering by showing that the problem remains PSpace-hard even if the query, the TBox, and the ABox are all fixed (and only the RBox varies); thus, we identify role inclusions as the only culprit for the problems' PSpace-hardness. Finally, we showed that positive, converse-free GXQs and CGXQs can be answered over OWL 2 EL knowledge bases in PTime and PSpace, respectively; this is interesting because Bienvenu et al. (2014) have showed that adding the converse operator makes the problem ExpTime-hard. Thus, at least from a theoretical perspective, positive, converse-free (C)GXQs seem to provide an adequate language for querying OWL 2 EL knowledge bases.

We see two main open problems for our future work. First, by drawing inspiration from the succinct encoding of role inclusions, we shall extend the combined approach by Stefanoni et al. (2013) to OWL 2 EL and thus obtain a practical algorithm. Second, as static query analysis is a fundamental task in query optimisation, we shall study the containment problem for graph queries under $\mathcal{ELRO}^+$ constraints.

Acknowledgements

The results in this article are an extension of the results that were published in preliminary form by Krötzsch et al. (2007) in the Proceedings of the 6th International Semantic Web Conference (ISWC 2007). This work was supported by Alcatel-Lucent; the EU FP7 project OPTIQUE; the EPSRC projects MASI^3, Score!, and DBOnto; and the DFG project DIAMOND (Emmy Noether grant KR 4381/1-1).

References

Anselmo, M., Giammarresi, D., & Varricchio, S. (2003). Finite automata and non-self-embedding grammars. In *Proceedings of the 7th International Conference on Im-*

plementation and Application of Automata, CIAA'02, pp. 47–56, Berlin, Heidelberg. Springer-Verlag.

Artale, A., Calvanese, D., Kontchakov, R., & Zakharyaschev, M. (2009). The *DL-Lite* family and relations. *J. Artif. Intell. Res. (JAIR)*, *36*, 1–69.

Baader, F., Brandt, S., & Lutz, C. (2005). Pushing the $\mathcal{EL}$ envelope. In Kaelbling, L. P., & Saffiotti, A. (Eds.), *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pp. 364–369, Edinburgh, UK. Morgan Kaufmann Publishers.

Baader, F., Brandt, S., & Lutz, C. (2008). Pushing the $\mathcal{EL}$ envelope further. In Clark, K., & Patel-Schneider, P. F. (Eds.), *In Proceedings of the OWLED 2008 DC Workshop on OWL: Experiences and Directions*.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., & Patel-Schneider, P. F. (Eds.). (2010). *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press. Paperback edition.

Baget, J.-F., Leclère, M., Mugnier, M.-L., & Salvat, E. (2011). On rules with existential variables: Walking the decidability line. *Artif. Intell.*, *175*(9-10), 1620–1654.

Barceló, P. (2013). Querying graph databases. In Hull, R., & Fan, W. (Eds.), *PODS*, pp. 175–188. ACM.

Barrett, C., Jacob, R., & Marathe, M. (2000). Formal-language-constrained path problems. *SIAM J. Comput.*, *30*(3), 809–837.

Bienvenu, M., Calvanese, D., Ortiz, M., & Simkus, M. (2014). Nested regular path queries in Description Logics. In *Proc. of the 14th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2014)*. AAAI Press.

Bienvenu, M., Ortiz, M., & Simkus, M. (2013). Conjunctive regular path queries in lightweight Description Logics. In Rossi, F. (Ed.), *IJCAI*. IJCAI/AAAI.

Calì, A., Gottlob, G., & Kifer, M. (2013). Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res. (JAIR)*, *48*, 115–174.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rodriguez-Muro, M., Rosati, R., Ruzzi, M., & Savo, D. F. (2011). The MASTRO system for Ontology-Based Data Access. *Semantic Web*, *2*(1), 43–53.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2006). Data complexity of query answering in Description Logics. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, pp. 260–270.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2007). Tractable reasoning and efficient query answering in Description Logics: The *DL-Lite* family. *J. Autom. Reasoning*, *39*(3), 385–429.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2013). Data complexity of query answering in Description Logics. *Artificial Intelligence*, *195*, 335–360.

Calvanese, D., De Giacomo, G., Lenzerini, M., & Vardi, M. Y. (2000). Containment of conjunctive regular path queries with inverse. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pp. 176–185.

Calvanese, D., Eiter, T., & Ortiz, M. (2009). Regular path queries in expressive Description Logics with nominals. In Boutilier, C. (Ed.), *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pp. 714–720.

Calvanese, D., Vardi, M. Y., De Giacomo, G., & Lenzerini, M. (2000). View-based query processing for regular path queries with inverse. In *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '00, pp. 58–66, New York, NY, USA. ACM.

Chandra, A. K., & Merlin, P. M. (1977). Optimal implementation of conjunctive queries in relational data bases. In Hopcroft, J. E., Friedman, E. P., & Harrison, M. A. (Eds.), *Proc. of the 9th annual ACM Symposium on Theory of Computing (STOC '77)*, pp. 77–90, Boulder, CO, USA. ACM Press.

Cruz, I. F., Mendelzon, A. O., & Wood, P. T. (1987). A graphical query language supporting recursion. *SIGMOD Rec.*, *16*(3), 323–330.

Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P. F., & Sattler, U. (2008). OWL 2: The next step for OWL. *J. Web Sem.*, *6*(4), 309–322.

De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rosati, R., Ruzzi, M., & Savo, D. F. (2012). MASTRO: A reasoner for effective Ontology-Based Data Access. In Horrocks, I., Yatskevich, M., & Jiménez-Ruiz, E. (Eds.), *ORE*, Vol. 858 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Eiter, T., Ortiz, M., & Simkus, M. (2012a). Conjunctive query answering in the Description Logic $\mathcal{SH}$ using knots. *J. Comput. Syst. Sci.*, *78*(1), 47–85.

Eiter, T., Ortiz, M., Simkus, M., Tran, T.-K., & Xiao, G. (2012b). Query rewriting for Horn-$\mathcal{SHIQ}$ plus rules. In Hoffmann, J., & Selman, B. (Eds.), *AAAI*. AAAI Press.

Fan, W. (2012). Graph pattern matching revised for social network analysis. In Deutsch, A. (Ed.), *ICDT*, pp. 8–21. ACM.

Geffert, V., Mereghetti, C., & Palano, B. (2010). More concise representation of regular languages by automata and regular expressions. *Information and computation*, *208*(4), 385–394.

Giese, M., Calvanese, D., Haase, P., Horrocks, I., Ioannidis, Y., Kllapi, H., Koubarakis, M., Lenzerini, M., Möller, R., Rodriguez-Muro, M., Özcep, O., Rosati, R., Schlatte, R., Schmidt, M., Soylu, A., & Waaler, A. (2013). Scalable end-user access to big data. In Akerkar, R. (Ed.), *Big Data Computing*. CRC Press.

Glimm, B., Lutz, C., Horrocks, I., & Sattler, U. (2008). Conjunctive query answering for the Description Logic $\mathcal{SHIQ}$. *J. Artif. Intell. Res. (JAIR)*, *31*, 157–204.

Gottlob, G., Manna, M., & Pieris, A. (2014). Polynomial combined rewritings for existential rules. In *Proc. of the 14th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2014)*. AAAI Press.

Gottlob, G., & Schwentick, T. (2012). Rewriting ontological queries into small nonrecursive datalog programs. In Brewka, G., Eiter, T., & McIlraith, S. A. (Eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press.

Grosof, B. N., Horrocks, I., Volz, R., & Decker, S. (2003). Description Logic Programs: Combining logic programs with Description Logic. In *Proceedings of the 12th international conference on World Wide Web*, pp. 48–57.

Gutierrez, C., Hurtado, C. A., Mendelzon, A. O., & Pérez, J. (2011). Foundations of semantic web databases. *J. Comput. Syst. Sci.*, *77*(3), 520–541.

Harel, D. (1984). Dynamic logic. In Gabbay, D., & Guenthner, F. (Eds.), *Handbook of Philosophical Logic Vol. II*, pp. 497–604. Reidel Publishing Company.

Harel, D., Tiuryn, J., & Kozen, D. (2000). *Dynamic Logic*. MIT Press, Cambridge, MA, USA.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2003). *Introduction to Automata Theory, Languages, and Computation - international edition (2. ed)*. Addison-Wesley.

Horrocks, I., Kutz, O., & Sattler, U. (2006). The even more irresistible $\mathcal{SROIQ}$. In Doherty, P., Mylopoulos, J., & Welty, C. A. (Eds.), *KR*, pp. 57–67. AAAI Press.

Horrocks, I., & Sattler, U. (2004). Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms. *Artificial Intelligence*, *160*(1–2), 79–104.

Johnson, D. S., & Klug, A. C. (1984). Testing containment of conjunctive queries under functional and inclusion dependencies. *J. Comput. Syst. Sci.*, *28*(1), 167–189.

Kazakov, Y. (2008). $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In Brewka, G., & Lang, J. (Eds.), *KR*, pp. 274–284. AAAI Press.

Kontchakov, R., Lutz, C., Toman, D., Wolter, F., & Zakharyaschev, M. (2011). The combined approach to Ontology-Based Data Access. In Walsh, T. (Ed.), *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pp. 2656–2661. IJCAI/AAAI.

Kostylev, E. V., Reutter, J. L., & Vrgoc, D. (2014). XPath for *DL-Lite* ontologies. In Bienvenu, M., Ortiz, M., Rosati, R., & Simkus, M. (Eds.), *Informal Proceedings of the 27th International Workshop on Description Logics, Vienna, Austria, July 17-20, 2014.*, Vol. 1193 of *CEUR Workshop Proceedings*, pp. 258–269. CEUR-WS.org.

Kozen, D. (1977). Lower bounds for natural proof systems. In *FOCS*, pp. 254–266. IEEE Computer Society.

Krötzsch, M. (2011). Efficient rule-based inferencing for OWL EL. In Walsh, T. (Ed.), *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*. AAAI Press/IJCAI. 2668–2673.

Krötzsch, M., Rudolph, S., & Hitzler, P. (2007). Conjunctive queries for a tractable fragment of OWL 1.1. In Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., & Cudré-Mauroux, P. (Eds.), *Proceedings of the 6th International Semantic Web Conference (ISWC'07)*, Vol. 4825 of *LNCS*, pp. 310–323. Springer.

Libkin, L., Martens, W., & Vrgoč, D. (2013). Querying graph databases with XPath. In Tan, W.-C., Guerrini, G., Catania, B., & Gounaris, A. (Eds.), *ICDT*, pp. 129–140. ACM.

Lutz, C. (2008). The complexity of conjunctive query answering in expressive Description Logics. In *Automated Reasoning*.

Lutz, C., Seylan, I., Toman, D., & Wolter, F. (2013). The combined approach to OBDA: Taming role hierarchies using filters. In Alani, H., Kagal, L., Fokoue, A., Groth, P. T., Biemann, C., Parreira, J. X., Aroyo, L., Noy, N. F., Welty, C., & Janowicz, K. (Eds.), *International Semantic Web Conference (1)*, Vol. 8218 of *Lecture Notes in Computer Science*, pp. 314–330. Springer.

Lutz, C., Toman, D., & Wolter, F. (2009). Conjunctive query answering in the Description Logic $\mathcal{EL}$ using a relational database system. In Boutilier, C. (Ed.), *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pp. 2070–2075.

Marnette, B. (2009). Generalized schema-mappings: From termination to tractability. In Paredaens, J., & Su, J. (Eds.), *PODS*, pp. 13–22. ACM.

Mora, J., Rosati, R., & Corcho, Ó. (2014). kyrie2: Query rewriting under extensional constraints in *ELHIO*. In Mika, P., Tudorache, T., Bernstein, A., Welty, C., Knoblock, C. A., Vrandecic, D., Groth, P. T., Noy, N. F., Janowicz, K., & Goble, C. A. (Eds.), *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, Vol. 8796 of *Lecture Notes in Computer Science*, pp. 568–583. Springer.

Ortiz, M., Calvanese, D., & Eiter, T. (2008). Data complexity of query answering in expressive Description Logics via tableaux. *J. Autom. Reasoning*, *41*(1), 61–98.

Ortiz, M., Rudolph, S., & Simkus, M. (2011). Query answering in the Horn fragments of the Description Logics $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$. In Walsh, T. (Ed.), *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pp. 1039–1044. IJCAI/AAAI.

Pérez, J., Arenas, M., & Gutierrez, C. (2010). nSPARQL: A navigational language for RDF. *Web Semant.*, *8*(4), 255–270.

Pérez-Urbina, H., Motik, B., & Horrocks, I. (2010). Tractable query answering and rewriting under Description Logic constraints. *J. Applied Logic*, *8*(2), 186–209.

Rodriguez-Muro, M., & Calvanese, D. (2012). High performance query answering over *DL-Lite* ontologies. In Brewka, G., Eiter, T., & McIlraith, S. A. (Eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press.

Rosati, R. (2007). On conjunctive query answering in $\mathcal{EL}$. In Calvanese, D., Franconi, E., Haarslev, V., Lembo, D., Motik, B., Turhan, A.-Y., & Tessaris, S. (Eds.), *Description Logics*, Vol. 250 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Rudolph, S., & Glimm, B. (2010). Nominals, inverses, counting, and conjunctive queries or: Why infinity is your friend!. *J. Artif. Intell. Res. (JAIR)*, *39*, 429–481.

Simančík, F. (2012). Elimination of complex rias without automata. In Kazakov, Y., Lembo, D., & Wolter, F. (Eds.), *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012*, Vol. 846 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., & Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *J. Web Sem.*, *5*(2), 51–53.

Stefanoni, G., Motik, B., & Horrocks, I. (2013). Introducing nominals to the combined query answering approaches for $\mathcal{EL}$. In desJardins, M., & Littman, M. L. (Eds.), *AAAI*. AAAI Press.

ter Horst, H. J. (2005). Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *Web Semantics: Science, Services and Agents on the World Wide Web*, *3*(2-3), 79–115.

Tsarkov, D., & Horrocks, I. (2006). FaCT++ Description Logic reasoner: System description. In Furbach, U., & Shankar, N. (Eds.), *IJCAR*, Vol. 4130 of *Lecture Notes in Computer Science*, pp. 292–297. Springer.

Urbani, J., van Harmelen, F., Schlobach, S., & Bal, H. E. (2011). QueryPIE: Backward reasoning for OWL Horst over very large knowledge bases. In Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N. F., & Blomqvist, E. (Eds.), *International Semantic Web Conference (1)*, Vol. 7031 of *Lecture Notes in Computer Science*, pp. 730–745. Springer.

Vardi, M. Y. (1982). The complexity of relational query languages (extended abstract). In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, STOC '82, pp. 137–146, New York, NY, USA. ACM.

Venetis, T., Stoilos, G., & Stamou, G. B. (2012). Incremental query rewriting for OWL 2 QL. In Kazakov, Y., Lembo, D., & Wolter, F. (Eds.), *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012*, Vol. 846 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Virgilio, R. D., Orsi, G., Tanca, L., & Torlone, R. (2012). NYAYA: A system supporting the uniform management of large sets of semantic data. In Kementsietsidis, A., & Salles, M. A. V. (Eds.), *IEEE 28th International Conference on Data Engineering (ICDE 2012), Washington, DC, USA (Arlington, Virginia), 1-5 April, 2012*, pp. 1309–1312. IEEE Computer Society.

Wessel, M. (2001). Obstacles on the Way to Qualitative Spatial Reasoning with Description Logics: Some Undecidability Results. In *Working Notes of the 2001 International Description Logics Workshop (DL-2001)*, Vol. 49. CEUR-WS.org.