PUBLICATIONS DE L'INSTITUT MATHÉMATIQUE
Nouvelle série, tome 95 (109) (2014), 133–147 DOI: 10.2298/PIM1409133M

ON THE COMPLEXITY OF (RESTRICTED) $\mathcal{ALCI}r$

Milenko Mosurović and Michael Zakharyaschev

Communicated by Žarko Mijajlović

ABSTRACT. We consider a new description logic $\mathcal{ALCI}r$ that extends $\mathcal{ALCI}$ with role inclusion axioms of the form $R \sqsubseteq QR_1 \dots R_m$ satisfying a certain regularity condition. We prove that concept satisfiability with respect to RBoxes in this logic is EXPTIME-hard. We then define a restriction $\mathcal{ALCI}r^-$ of $\mathcal{ALCI}r$ and show that concept satisfiability with respect to RBoxes in $\mathcal{ALCI}r^-$ is PSPACE-complete.

1. Introduction

Description logics (DLs, for short) are well-known knowledge representation formalisms in AI [**2**] and the Semantic Web, underlining the Web Ontology Language OWL 2.[1] The language of DLs is built around concept names (unary predicates) and role names (binary predicates) using various constructs such as the Booleans, universal and existential restrictions, taking the inverse of a role, etc. The resulting complex concepts and roles are used to form knowledge base (or ontology) axioms. For example, the axiom $\mathit{Parent} \equiv \mathit{Person} \sqcap \exists \mathit{hasChild}.\,\mathit{Person}$ defines a parent as a person who has child, which is also a person, while $\mathit{ancestor} \circ \mathit{ancestor} \sqsubseteq \mathit{ancestor}$ says that *ancestor* is a transitive role. An important feature of standard DLs is that reasoning problems such as concept and role satisfiability with respect to knowledge bases or instance checking are decidable and reasonably scalable in practice; moreover, there are a number of highly optimized DL reasoners for performing those reasoning tasks.[2]

In recent years, a considerable interest has been attracted to DLs with complex role inclusion axioms of the form $R_1 \circ \cdots \circ R_n \sqsubseteq S_1 \circ \cdots \circ S_k$, where the R_i and S_j are role names or their inverses. Such axioms are required, for instance, in multiple use cases in the life sciences domain and ontological product modelling and collaborative design [**14, 5, 3, 12**]. In fact, complex role inclusion axioms were already used in the original KL-ONE terminological language [**4**], where they were

2010 *Mathematics Subject Classification:* 68T27, 03B70, 68T30, 68W40, 68Q25.

[1]http://www.w3.org/TR/owl2-overview/

[2]See, e.g., http://en.wikipedia.org/wiki/Description_logic#Tools

called 'role-value-maps' and could be applied to certain individuals. However, it turned out [**16**] that KL-ONE was undecidable. One way to defeat undecidability is to restrict the right-hand side of role inclusions to a single role and impose on them a certain regularity condition, which is done in the DL $\mathcal{SROIQ}$ underlying OWL 2 [**9, 5**] (see also [**1, 7, 10, 8, 11**]).

A decidable DL, $\mathcal{SR}^+\mathcal{OIQ}$, with role inclusion axioms that allow a non-trivial right-hand side (and also satisfy a regularity condition) has been recently constructed in [**13**]. However, the exact complexity of reasoning with this logic is still unknown. To understand the impact of such role inclusions, we here consider the fragment $\mathcal{ALCI}r$ of $\mathcal{SR}^+\mathcal{OIQ}$ which extends the basic DL $\mathcal{ALCI}$ with axioms of the form $R \sqsubseteq QR_1 \dots R_m$ satisfying the same regularity condition as $\mathcal{SR}^+\mathcal{OIQ}$. We prove, in Section 3, that concept satisfiability with respect to RBoxes (finite sets of role inclusion axioms) in $\mathcal{ALCI}r$ is ExpTime-hard. However, we have not managed yet to obtain a matching upper bound. Instead, in Section 4, we develop a PSpace tableau-based decision algorithm for a fragment of $\mathcal{ALCI}r$ which imposes further restrictions on role inclusion axioms.

2. Description Logic $\mathcal{ALCI}r$

We begin by formally defining the syntax and semantics of the description logic $\mathcal{ALCI}r$. The *alphabet* of $\mathcal{ALCI}r$ consists of two countably infinite and disjoint sets $\mathcal{N}_C$ and $\mathcal{N}_R$ of *concept names* and *role names*, respectively. This alphabet is interpreted in structures, or *interpretations*, of the form $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I} \neq \emptyset$ is the *domain* of interpretation and $\cdot^\mathcal{I}$ an *interpretation function* that assigns to every $A \in \mathcal{N}_C$ a subset $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and to every $R \in \mathcal{N}_R$ a binary relation $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$.

We now introduce the role and concept constructs that are available in $\mathcal{ALCI}r$, together with their interpretations. For each role name $R \in \mathcal{N}_R$, the *inverse* R^- of R is interpreted by the relation

$$(R^-)^\mathcal{I} = \{(y, x) \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid (x, y) \in R^\mathcal{I}\}.$$

We call role names and their inverses *basic roles*, set $\mathcal{N}_R^- = \mathcal{N}_R \cup \{R^- \mid R \in \mathcal{N}_R\}$ and write $\mathrm{rn}(R) = \mathrm{rn}(R^-) = R$, for $R \in \mathcal{N}_R$. We define an $\mathcal{ALCI}r$*-role* as a *chain* $R_1 \dots R_n$ of basic roles R_i and interpret it by taking

$$(R_1 \dots R_n)^\mathcal{I} = R_1^\mathcal{I} \circ \dots \circ R_n^\mathcal{I},$$

where $\circ$ denotes the composition of binary relations. We also define a function $\mathrm{inv}(\cdot)$ on role chains by $\mathrm{inv}(R_1 \cdots R_n) = \mathrm{inv}(R_n) \cdots \mathrm{inv}(R_1)$, where $\mathrm{inv}(R) = R^-$ and $\mathrm{inv}(R^-) = R$, for $R \in \mathcal{N}_R$.

$\mathcal{ALCI}r$*-concepts*, C, are defined by the following grammar, where $A \in \mathcal{N}_C$ and R is a basic role:

$$C \quad ::= \quad A \quad | \quad \bot \quad | \quad \top \quad | \quad \neg C \quad | \quad C_1 \sqcap C_2 \quad | \quad C_1 \sqcup C_2 \quad | \quad \exists R.C \quad | \quad \forall R.C.$$

The interpretation of these concepts is defined as follows:

$$\top^\mathcal{I} = \Delta^\mathcal{I}, \qquad \bot^\mathcal{I} = \emptyset, \qquad (\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I},$$

$$(C_1 \sqcap C_2)^\mathcal{I} = C_1^\mathcal{I} \cap C_2^\mathcal{I}, \qquad (C_1 \sqcup C_2)^\mathcal{I} = C_1^\mathcal{I} \cup C_2^\mathcal{I},$$

$$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y \in C^{\mathcal{I}}\ (x, y) \in R^{\mathcal{I}}\},$$
$$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\, ((x, y) \in R^{\mathcal{I}} \to y \in C^{\mathcal{I}})\}.$$

An *RBox*, $\mathcal{R}$, is a finite set of *role inclusions axiom* (RIs) of the form

$$R \sqsubseteq QR_1 \dots R_m, \qquad m \geqslant 1,$$

where $R, Q, R_1, \dots, R_m$ are basic roles satisfying the following *regularity condition*: the transitive closure of the relation induced by $\mathrm{rn}(R) \prec \mathrm{rn}(Q)$, for all such RIs in $\mathcal{R}$, is a strict partial order on the set of role names. We denote this strict partial order by $\prec_{\mathcal{R}}$. If ϱ_i is a role, $i = 1, 2$, then $\varrho_1 \sqsubseteq \varrho_2$ is satisfied in $\mathcal{I}$ if $\varrho_1^{\mathcal{I}} \subseteq \varrho_2^{\mathcal{I}}$. We say that an RBox $\mathcal{R}$ is *satisfiable* if there exists an interpretation $\mathcal{I}$ satisfying all the members of $\mathcal{R}$. In this case we write $\mathcal{I} \models \mathcal{R}$ and call $\mathcal{I}$ a *model* of $\mathcal{R}$.

The main reasoning problem we are concerned with in this paper is *concept satisfiability with respect to an RBox*: given an $\mathcal{ALCI}r$ concept C and an RBox $\mathcal{R}$, decide whether there is a model $\mathcal{I}$ of $\mathcal{R}$ such that $C^{\mathcal{I}} \neq \emptyset$.

For a concept C, we denote by $\mathrm{role}(C)$ the set of all basic roles R such that at least one of R or $\mathrm{inv}(R)$ occurs in C; $\mathrm{role}(C, \mathcal{R})$ contains those basic roles and their inverses that occur in C or $\mathcal{R}$.

We assume that all concepts are in *negation normal form* (NNF). In particular, when we write $\neg C$, for a concept C, we actually mean the NNF of $\neg C$. Denote by $\mathrm{con}(C)$ the smallest set that contains C and is closed under sub-concepts and $\neg$.

The DL $\mathcal{ALCI}r$ defined above is an extension of the well-known DL $\mathcal{ALCI}$ [**2**] with regular RBoxes. Thus, concept satisfiability w.r.t. RBoxes in $\mathcal{ALCI}r$ is PSpace-hard. On the other hand, $\mathcal{ALCI}r$ is a fragment of the decidable DL $\mathcal{SR}^{+}\mathcal{OIQ}$ [**13**] the precise complexity of reasoning in which is still unknown.

3. ExpTime-Hardness

In this section we establish an ExpTime lower bound for the complexity of concept satisfiability w.r.t. RBoxes in $\mathcal{ALCI}r$. The following example illustrates the reduction used in the proof.

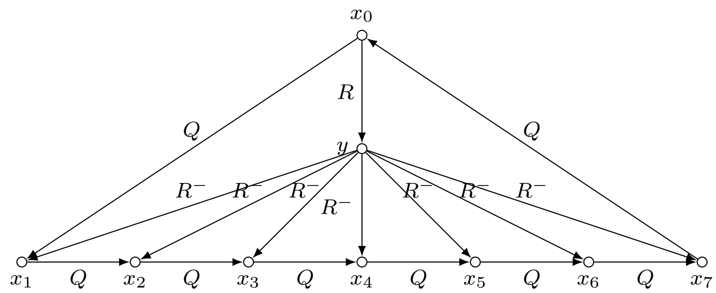
Example 3.1. Let $\mathcal{R} = \{R \sqsubseteq QR\}$. Consider the concept

$$C_0 = (\neg A_{n-1} \sqcap \dots \sqcap \neg A_0) \sqcap \exists R.\top \sqcap C' \sqcap C'',$$

where

$$C' = \prod_{k=0}^{n-1} \forall R.\forall R^{-}.\Bigg(\bigg(\prod_{i=0}^{k-1} A_i\bigg) \Rightarrow \big((A_k \Rightarrow \forall Q.\neg A_k) \sqcap (\neg A_k \Rightarrow \forall Q.A_k)\big)\Bigg),$$
$$C'' = \prod_{k=0}^{n-1} \forall R.\forall R^{-}.\Bigg(\bigg(\bigsqcup_{i=0}^{k-1} \neg A_i\bigg) \Rightarrow \big((A_k \Rightarrow \forall Q.A_k) \sqcap (\neg A_k \Rightarrow \forall Q.\neg A_k)\big)\Bigg)$$

and $C \Rightarrow D$ is an abbreviation for $\neg C \sqcup D$. It is not hard to see that C_0 is of size $O(n^2)$ and that every model of $\mathcal{R}$ satisfying C_0 has a Q-path of length 2^n. For $n = 3$, such a model $\mathcal{I}$ is shown below, where $x_0 \in (\neg A_2 \sqcap \neg A_1 \sqcap \neg A_0)^{\mathcal{I}}$, $x_1 \in (\neg A_2 \sqcap \neg A_1 \sqcap A_0)^{\mathcal{I}}$, $x_2 \in (\neg A_2 \sqcap A_1 \sqcap \neg A_0)^{\mathcal{I}}, \dots, x_7 \in (A_2 \sqcap A_1 \sqcap A_0)^{\mathcal{I}}$.



In other words, the concepts $A_{n-1}, \ldots, A_0$ encode the bits of an n-bit binary counter, with A_0 representing the least significant bit and A_{n-1} the most significant one.

THEOREM 3.1. *Concept satisfiability w.r.t. $\mathcal{ALCI}r$-RBoxes is* EXPTIME-*hard.*

PROOF. The proof is by encoding computations of alternating Turing machines with a polynomial tape. Recall [**6**] that an *alternating Turing machine* (ATM) is a quadruple of the form $\mathcal{M} = (Q, \Sigma, q_0, \delta)$, where $Q = Q_\exists \uplus Q_\forall \uplus \{q_a\} \uplus \{q_r\}$ is a finite set of *states* containing *existential states* $Q_\exists$, *universal states* $Q_\forall$, an *accepting state* q_a and a *rejecting state* q_r; $\Sigma = \{a_0, \ldots, a_{n-1}, \boldsymbol{b}\}$ is a *tape alphabet* with a special symbol $\boldsymbol{b}$ for 'blank'; $q_0 \in Q_\exists \uplus Q_\forall$ is an *initial state*; and

$$\delta \colon (Q \smallsetminus \{q_a, q_r\}) \times \Sigma \to (Q \times \Sigma \times \{l, r\})^2$$

is a *transition function* that, for any $q \in Q \smallsetminus \{q_a, q_r\}$ and $a \in \Sigma$, gives two alternative transitions. If $\delta(q, a) = ((q_1, a_1, d_1), (q_2, a_2, d_2))$, then we write $\delta_i(q, a) = (q_i, a_i, d_i)$, for $i = 1, 2$.

A *configuration* of an ATM $\mathcal{M}$ is a word wqw', where $q \in Q$ and $w, w' \in \Sigma^*$; it represents the situation when the tape contains the word ww', the machine is in state q, with its head scanning the first symbol of w'. The *successor configurations* of wqw' are defined in the usual way in terms of the transition function δ (where l and r instruct the head to move one cell to the left or, respectively, to the right). A *halting configuration* is of the form wqw' with $q \in \{q_a, q_r\}$.

A *computation path* of $\mathcal{M}$ on a word $w \in \Sigma^*$ is a sequence of configurations $c_1, c_2, \ldots$ such that $c_1 = q_0 w$ and, for $i \geqslant 1$, c_{i+1} is a successor configuration of c_i. We assume that there is a polynomial function p such that $\mathcal{M}$ uses at most $p(m)$ tape cells on any input of length m and that every computation path of $\mathcal{M}$ is of length $\leqslant 2^{p(m)}$. A halting configuration is *accepting* if it is of the form $wq_a w'$. For a non-halting configuration $c = wqw'$, we define (inductively) c to be *accepting* if either $q \in Q_\exists$ and at least one of the successor configurations of c is accepting, or $q \in Q_\forall$ and both the successor configurations are accepting. Finally, we say that $\mathcal{M}$ *accepts* an input w if the initial configuration $q_0 w$ is accepting. We use $L(\mathcal{M})$ to denote the language accepted by $\mathcal{M}$, that is, $L(\mathcal{M}) = \{w \in \Sigma^* \mid \mathcal{M} \text{ accepts } w\}$. It is well known [**6**] that the *acceptance problem* for such ATMs is EXPTIME-complete.

Now, given an ATM $\mathcal{M}$ and a word $w = x_0 \cdots x_{n-1} \in \Sigma^*$, we construct a concept C_0 and an RBox $\mathcal{R}$ in $\mathcal{ALCI}r$ such that $w \in L(\mathcal{M})$ iff C_0 is satisfiable w.r.t. $\mathcal{R}$. To achieve this, we use Example 3.1 to represent successive configurations of $\mathcal{M}$ by individuals of models of $\mathcal{R}$ satisfying C_0 and arrange them in a binary tree-like structure with branching factor 2 and depth $\leqslant 2^{p(n)}$.

Consider the RBox

$$\mathcal{R} = \{R \sqsubseteq Q_1 R, \quad R \sqsubseteq Q_2 R\}.$$

We use the role name Q_j, $j = 1, 2$, to connect two successive configurations c and c' such that c' is obtained from c by means of the jth component of the pair $\delta(q, a)$ that is applicable to c. We require the following concept names:

- A to say that a configuration is accepting;
- B_q, for $q \in Q$, to say that the current state is q;
- H_i, for $i \leqslant p(n)$, to say that the head is currently scanning cell i;
- D_a^i, for $a \in \Sigma$ and $i \leqslant p(n)$, to say that the current symbol in the ith cell is a;
- A_i, for $i < p(n)$, to encode the binary counter;
- E to say that an object is a Q_1-successor of its parent;
- $F \equiv (A_{p(n)-1} \sqcap \cdots \sqcap A_0)$ to say that an object is a leaf.

To construct the required binary tree-like structure, we use the following concept C_T, where $C \Leftrightarrow D$ stands for $(C \Rightarrow D) \sqcap (D \Rightarrow C)$:

$$C_T = C^{\text{zero}} \sqcap C^F \sqcap \exists R.\top \sqcap \prod_{j=1}^{2} (C'_j \sqcap C''_j \sqcap C'''_j),$$

where

$$\begin{aligned}
C^{\text{zero}} &= \neg A_{p(n)-1} \sqcap \cdots \sqcap \neg A_0, \\
C^F &= \forall R.\forall R^-.\big(F \Leftrightarrow (A_{p(n)-1} \sqcap \cdots \sqcap A_0)\big), \\
C'_j &= \prod_{k=0}^{p(n)-1} \forall R.\forall R^-.(F \sqcup C^{\text{invert}}_{k,j}), \\
C^{\text{invert}}_{k,j} &= \left(\prod_{i=0}^{k-1} A_i\right) \Rightarrow \Big((A_k \Rightarrow \forall Q_j.\neg A_k) \sqcap (\neg A_k \Rightarrow \forall Q_j.A_k)\Big), \\
C''_j &= \prod_{k=0}^{p(n)-1} \forall R.\forall R^-.(F \sqcup C^{\text{retain}}_{k,j}), \\
C^{\text{retain}}_{k,j} &= \left(\bigsqcup_{i=0}^{k-1} \neg A_i\right) \Rightarrow \Big((A_k \Rightarrow \forall Q_j.A_k) \sqcap (\neg A_k \Rightarrow \forall Q_j.\neg A_k)\Big), \\
C'''_1 &= \forall R.\forall R^-.\forall Q_1.E, \qquad C'''_2 = \forall R.\forall R^-.\forall Q_2.\neg E.
\end{aligned}$$

It is not hard to see that there is a model of $\mathcal{R}$ satisfying C_T that contains the full binary tree of depth $2^{p(n)} - 1$ every non-leaf node of which has distinct Q_1-and a Q_2-successors.

We now construct the concept C_0 by giving its subconcepts. The concept C^{ini} defines the initial configuration. $C^l_{(q,i,a)}$ and $C^r_{(q,i,a)}$ describe the movements of the head to the left and right. C_{nh} makes sure that the symbols that are not in the active cell do not change. C_{dfh}, C_{dfq} and C_{dfs} ensure uniqueness of the head position, current state and symbol in the active cell. C_{acc} describes accepting configurations using concepts $C_\forall$ and $C_\exists$. These concepts are defined as follows:

$$
\begin{aligned}
C^{\mathrm{ini}} &= B_{q_0} \sqcap H_0 \sqcap \Big(\bigsqcap_{i=0}^{n-1} D^i_{x_i}\Big) \sqcap \Big(\bigsqcap_{i=n}^{p(n)} D^i_{\boldsymbol{b}}\Big),\\
C^r_{(q,i,a)} &= (B_q \sqcap H_i \sqcap D^i_a) \Rightarrow \Big(\bigsqcap_{t\in\{1,2\},\, \delta_t(q,a)=(q',a',r)} \forall Q_t.(B_{q'} \sqcap H_{i+1} \sqcap D^i_{a'})\Big),\\
C^l_{(q,i,a)} &= (B_q \sqcap H_i \sqcap D^i_a) \Rightarrow \Big(\bigsqcap_{t\in\{1,2\},\, \delta_t(q,a)=(q',a',l)} \forall Q_t.(B_{q'} \sqcap H_{i-1} \sqcap D^i_{a'})\Big),\\
C_{\mathrm{nh}} &= \bigsqcap_{a\in\Sigma,\, 0\leqslant i,j\leqslant p(n),\, i\neq j} \forall R.\forall R^-.\big((H_j \sqcap D^i_a) \Rightarrow (\forall Q_1.D^i_a \sqcap \forall Q_2.D^i_a)\big),\\
C_{\mathrm{dfh}} &= \bigsqcap_{0\leqslant j<i\leqslant p(n)} \forall R.\forall R^-.(\neg H_j \sqcup \neg H_i),\\
C_{\mathrm{dfq}} &= \bigsqcap_{q,q'\in Q,\, q\neq q'} \forall R.\forall R^-.(\neg B_q \sqcup \neg B_{q'}),\\
C_{\mathrm{dfs}} &= \bigsqcap_{a,b\in\Sigma,\, a\neq b,\, 0\leqslant i\leqslant p(n)} \forall R.\forall R^-.(\neg D^i_a \sqcup \neg D^i_b),\\
C_\forall &= \bigsqcup_{q\in Q_\forall,\, a\in\Sigma,\, 0\leqslant i\leqslant p(n)} \big(B_q \sqcap H_i \sqcap D^i_a \sqcap \forall Q_1.A \sqcap \forall Q_2.A\big),\\
C_\exists &= \bigsqcup_{q\in Q_\exists,\, a\in\Sigma,\, 0\leqslant i\leqslant p(n)} \big(B_q \sqcap H_i \sqcap D^i_a \sqcap (\forall Q_1.A \sqcup \forall Q_2.A)\big),\\
C_{\mathrm{acc}} &= A \sqcap \forall R.\forall R^-.\big((B_{q_a} \sqcup C_\forall \sqcup C_\exists) \Leftrightarrow A\big).
\end{aligned}
$$

Finally, we set

$$
\begin{aligned}
C_0 = C_T \sqcap C^{\mathrm{ini}} \sqcap C_{\mathrm{acc}} \sqcap C_{\mathrm{dfh}} \sqcap C_{\mathrm{dfq}} \sqcap C_{\mathrm{dfs}} \sqcap C_{\mathrm{nh}} \sqcap \\
\bigsqcap_{q\in Q,\, a\in\Sigma,\, 0\leqslant i\leqslant p(n)-1} \forall R.\forall R^-.C^r_{(q,i,a)} \sqcap \bigsqcap_{q\in Q, a\in\Sigma, 1\leqslant i\leqslant p(n)} \forall R.\forall R^-.C^l_{(q,i,a)}.
\end{aligned}
$$

The size of C_0 is $O(p(n)^2)$, and one can check that C_0 is satisfiable w.r.t. $\mathcal{R}$ iff the ATM $\mathcal{M}$ accepts input w. □

The precise complexity of concept satisfiability w.r.t. $\mathcal{ALCI}r$ RBoxes is still unknown. As shown by Example 3.1 and the proof above, models of $\mathcal{R}$ satisfying C_0 can be of exponential width. The tableau algorithm for $\mathcal{SR}^+\mathcal{OIQ}$ from [**13**] indicates that, for this more powerful language, we can expect multiple exponential blowups of the depth. However, we hope that this does not happen for $\mathcal{ALCI}r$ and conjecture that concept satisfiability w.r.t. $\mathcal{ALCI}r$ RBoxes is ExpTime-complete. Some grounds for this conjecture will be given in the next section, where we define a non-trivial fragment of $\mathcal{ALCI}r$ for which this problem turns out to be PSpace-complete.

4. Description Logic $\mathcal{ALCI}r^-$

Let $\mathcal{R}$ be an $\mathcal{ALCI}r$ RBox and C_0 an $\mathcal{ALCI}r$ concept. To simplify presentation, and without loss of generality, we can assume that all RIs are of the form $R \sqsubseteq QP$. Let $\mathcal{R} = \{\boldsymbol{r}_i \mid i = 1, \ldots, l\}$, where $\boldsymbol{r}_i = (R_i \sqsubseteq Q_i P_i)$, for $i = 1, \ldots, l$. We denote by $\mathcal{ALCI}r^-$ the fragment of $\mathcal{ALCI}r$ in which RBoxes are such that no role name $\mathrm{rn}(P_i)$, for $i = 1, \ldots, l$, occurs on the left-hand side of RIs (that is, $\mathrm{rn}(P_i) \neq \mathrm{rn}(R_j)$ for any $i, j \leqslant l$). Our aim in this section is to develop a tableau algorithm for this DL that works in the polynomial space. This algorithm can be obtained by properly modifying the tableau algorithm for $\mathcal{SR}^+\mathcal{OIQ}$ given in [**13**]. So, we begin by reminding the reader of the basic definitions and notation used [**13**].

4.1. Quasi-Concepts. First we define a set $\boldsymbol{qc}(C_0, \mathcal{R})$ the elements of which are called *quasi-concepts* (for C_0 w.r.t. $\mathcal{R}$); we need them in labels for tableau nodes. In the definition of $\boldsymbol{qc}(C_0, \mathcal{R})$, we require a dependency relation $\lhd$ on $\mathcal{R}$, which is defined by taking $\boldsymbol{r}_i \lhd \boldsymbol{r}_j$ if $\mathrm{rn}(R_i) = \mathrm{rn}(Q_j)$. The following lemma shows that the transitive closure of $\lhd$ is acyclic:

LEMMA 4.1. [**13**] (i) *If $\boldsymbol{r}_i \lhd \boldsymbol{r}_j$, then $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$ does not hold.*
(ii) *If $\boldsymbol{r}_{i_1} \lhd \boldsymbol{r}_{i_2}$ and $\boldsymbol{r}_{i_2} \lhd \boldsymbol{r}_{i_3}$, then $\boldsymbol{r}_{i_3} \lhd \boldsymbol{r}_{i_1}$ does not hold.*

For a set $\Sigma \subseteq \mathrm{con}(C_0)$ and a basic role P, we set $\Sigma|_P^\forall = \{C \mid \forall P.C \in \Sigma\}$. Sometimes it will be convenient to write $\boldsymbol{qc}(\boldsymbol{r}_0)$ in place of $\mathrm{con}(C_0)$ and assume that $\boldsymbol{r}_0 \lhd \boldsymbol{r}_i$, for $1 \leqslant i \leqslant l$. Now, assuming that $\boldsymbol{qc}(\boldsymbol{r}_j)$ is defined for every $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$ with $0 \leqslant i, j \leqslant l$, we define $\boldsymbol{qc}(\boldsymbol{r}_i)$ to be the set of all $\forall R_i.\exists P_i^-.(t^r, t^\forall, t^-)$ such that

$$t^r = Q_i, \qquad t^\forall \subseteq \bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)|_{t^r}^\forall, \qquad t^- \subseteq \bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)|_{\mathrm{inv}(t^r)}^\forall.$$

For $\Sigma(\boldsymbol{r}_i) \subseteq \boldsymbol{qc}(\boldsymbol{r}_i)$ and a basic role P, let

$$\Sigma(\boldsymbol{r}_i)|_P^\forall = \big\{\exists P_i^-.(t^r, t^\forall, t^-) \mid \forall R_i.\exists P_i^-.(t^r, t^\forall, t^-) \in \Sigma(\boldsymbol{r}_i) \text{ and } P = R_i\big\}.$$

Finally, we set $\boldsymbol{qc}(C_0, \mathcal{R}) = \bigcup_{i=0}^l \boldsymbol{qc}(\boldsymbol{r}_i)$, and, for $\Sigma \subseteq \boldsymbol{qc}(C_0, \mathcal{R})$ and a basic role P,

$$\Sigma|_P^\forall = \bigcup_{j=0}^{l} (\Sigma \cap \boldsymbol{qc}(\boldsymbol{r}_j))|_P^\forall.$$

For $\Sigma \subseteq \boldsymbol{qc}(C_0, \mathcal{R})$ and $1 \leqslant i \leqslant l$, let $\Xi(\boldsymbol{r}_i, \Sigma) = \exists P_i^-.(t^r, t^\forall, t^-)$, where $t^r = Q_i$, $t^\forall = \Sigma|_{t^r}^\forall$, $t^- = \Sigma \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{\mathrm{inv}(t^r)}^\forall$.

4.2. Tableaux and Concept Tableaux. A *tableau* for C_0 w.r.t. $\mathcal{R}$ is a structure of the form $\boldsymbol{T} = (\boldsymbol{S}, \ell, \mathcal{E})$, where $\boldsymbol{S}$ is a nonempty set, $\ell\colon \boldsymbol{S} \to 2^{\boldsymbol{qc}(C_0, \mathcal{R})}$ and $\mathcal{E}\colon \mathrm{role}(C_0, \mathcal{R}) \to 2^{\boldsymbol{S} \times \boldsymbol{S}}$ are such that the following conditions hold:

- p1: $C_0 \in \ell(u_0)$, for some $u_0 \in \boldsymbol{S}$,
- p2: if $C \in \ell(u)$, then $\neg C \notin \ell(u)$, where C is a concept name,
- p3: $\top \in \ell(u)$ and $\bot \notin \ell(u)$ for any u,
- p4: if $(C_1 \sqcap C_2) \in \ell(u)$, then $C_1 \in \ell(u)$ and $C_2 \in \ell(u)$,
- p5: if $(C_1 \sqcup C_2) \in \ell(u)$, then $C_1 \in \ell(u)$ or $C_2 \in \ell(u)$,
- p6: if $\exists R.C \in \ell(u)$, then there is some $v \in \boldsymbol{S}$ with $(u, v) \in \mathcal{E}(R)$ and $C \in \ell(v)$,

p7: $(u,v) \in \mathcal{E}(R)$ iff $(v,u) \in \mathcal{E}(\mathrm{inv}(R))$,
p8: if $\forall R.C \in \ell(u)$ and $(u,v) \in \mathcal{E}(R)$, then $C \in \ell(v)$,
p9: if $\forall R.\mathfrak{C} \in \ell(u)$ and $(u,v) \in \mathcal{E}(R)$, then $\mathfrak{C} \in \ell(v)$,
p10: $\forall R_i.\mathfrak{C} \in \ell(u)$, where $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \ell(u))$, for all $u \in \boldsymbol{S}$ and $1 \leqslant i \leqslant l$,
p11: if $\exists P.(t^r, t^\forall, t^-) \in \ell(u)$, then there is v with $(u,v) \in \mathcal{E}(P)$, $t^\forall \subseteq \ell(v)$ and $\ell(v)|^\forall_{\mathrm{inv}(t^r)} \subseteq t^-$.

A *concept tableau* for C_0 w.r.t. $\mathcal{R}$ is a structure of the form $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}, \mathcal{E})$, where $\boldsymbol{S}$ is nonempty set, $\mathfrak{c}\colon \boldsymbol{S} \to 2^{\mathrm{con}(C_0)}$ and $\mathcal{E}\colon \mathrm{role}(C_0, \mathcal{R}) \to 2^{\boldsymbol{S}\times\boldsymbol{S}}$ are such that:

a: conditions (p1)–(p8) hold for ℓ replaced by $\mathfrak{c}$;
b: if $(u,v) \in \mathcal{E}(R_i)$, then $(u,v) \in \mathcal{E}(Q_i) \circ \mathcal{E}(P_i)$.

THEOREM 4.1. *Suppose C_0 is an $\mathcal{ALCIr}$ concept and $\mathcal{R}$ an $\mathcal{ALCIr}$ RBox. Then the following conditions are equivalent:*

(a) *C_0 is satisfiable w.r.t. $\mathcal{R}$;*
(b) *there exists a tableau for C_0 w.r.t. $\mathcal{R}$;*
(c) *there exists a concept tableau for C_0 w.r.t. $\mathcal{R}$.*

PROOF. The equivalence (a) $\Leftrightarrow$ (b) follows from [**13**].

(b) $\Rightarrow$ (c). Let $\boldsymbol{T} = (\boldsymbol{S}, \ell, \mathcal{E})$ be a tableau for C_0 w.r.t. $\mathcal{R}$. Define a concept tableau $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}')$ by taking $\boldsymbol{S}' = \boldsymbol{S}$, $\mathfrak{c}(u) = \ell(u) \cap \mathrm{con}(C_0)$ for all $u \in \boldsymbol{S}$. For a role name R, we define $\mathcal{E}'(R)$, by induction on $\prec_{\mathcal{R}}$, as follows:

$$\begin{aligned}\mathcal{E}'(R) = \mathcal{E}(R) & \\ &\cup \bigcup_{\{i \mid R = Q_i\}} \{(u,v) \mid ar(R,u,v) \ \&\ \exists z\, ((u,z) \in \mathcal{E}'(R_i) \wedge (v,z) \in \mathcal{E}(P_i))\} \\ &\cup \bigcup_{\{i \mid R = \mathrm{inv}(Q_i)\}} \{(u,v) \mid ar(R,u,v) \ \&\ \exists z\, ((v,z) \in \mathcal{E}'(R_i) \wedge (u,z) \in \mathcal{E}(P_i))\}.\end{aligned}$$

Here $ar(R,u,v)$ is an abbreviation for $\ell(u)|^\forall_R \subseteq \ell(v)$ and $\ell(v)|^\forall_{\mathrm{inv}(R)} \subseteq \ell(u)$. We also set $\mathcal{E}'(\mathrm{inv}(R)) = \{(u,v) \mid (v,u) \in \mathcal{E}'(R)\}$. It is easy to see that $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}')$ is concept tableau for C_0 w.r.t. $\mathcal{R}$.

(c) $\Rightarrow$ (b) Let $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}')$ be a concept tableau for C_0 w.r.t. $\mathcal{R}$. Construct a tableau $\boldsymbol{T} = (\boldsymbol{S}, \ell, \mathcal{E})$ by taking $\boldsymbol{S} = \boldsymbol{S}'$, $\mathcal{E} = \mathcal{E}'$ and defining ℓ as follows. First, we define, by induction on $\lhd$, auxiliary sets $\ell'(u, \boldsymbol{r})$, where $\boldsymbol{r}$ is an RI. Recalling that $\boldsymbol{r}_0 \lhd \boldsymbol{r}_i$ for all i, $1 \leqslant i \leqslant l$, we set $\ell'(u, \boldsymbol{r}_0) = \mathfrak{c}(u)$. Then, assuming that $\ell'(u, \boldsymbol{r}')$ is defined for every $\boldsymbol{r}' \lhd \boldsymbol{r_i}$, we set

$$\begin{aligned}\ell'(u, \boldsymbol{r}_i) = \big\{\forall R_i.\mathfrak{C} \mid \mathfrak{C} = \Xi\big(\boldsymbol{r}_i, \textstyle\bigcup_{\boldsymbol{r}' \lhd \boldsymbol{r_i}} \ell'(u, \boldsymbol{r}')\big)\big\} & \\ \cup \big\{\mathfrak{C} \mid \exists v \in \boldsymbol{S},\ (v,u) \in \mathcal{E}'(R_i),\ \mathfrak{C} = \Xi\big(\boldsymbol{r}_i, \textstyle\bigcup_{\boldsymbol{r}' \lhd bsr_i} \ell'(v, \boldsymbol{r}')\big)\big\}&.\end{aligned}$$

Finally, we set $\ell(u) = \bigcup_{j=0}^{l} \ell'(u, \boldsymbol{r}_j)$. One can show now that $\boldsymbol{T}$ is a tableau for C_0 w.r.t. $\mathcal{R}$. $\square$

4.3. Completion Trees. We are now in a position to define a tableau algorithm for $\mathcal{ALCIr}$. For $\mathcal{ALCIr}$ concepts C_0 and RBoxes $\mathcal{R}$, the algorithm works on completion trees similarly to the algorithms of [**2, 13**]. To present it, we require some additional notation. For RI $\boldsymbol{r}_i = (R_i \sqsubseteq Q_i P_i)$ for $1 \leqslant i \leqslant l$, let

$$\boldsymbol{qc}^{+}(\boldsymbol{r}_i) = \bigl\{\forall Q_i.\mathfrak{C} \mid \forall Q_i.\mathfrak{C} \in \textstyle\bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)\bigr\},$$
$$\boldsymbol{qc}^{-}(\boldsymbol{r}_i) = \bigl\{\mathfrak{C} \mid \forall\,\mathrm{inv}(Q_i).\mathfrak{C} \in \textstyle\bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)\bigr\},$$
$$\boldsymbol{qc}^{*}(\boldsymbol{r}_i) = \boldsymbol{qc}^{+}(\boldsymbol{r}_i) \cup \boldsymbol{qc}^{-}(\boldsymbol{r}_i).$$

The set $\boldsymbol{qc}^*(\boldsymbol{r}_i)$ of quasi-concepts is to be guessed by the algorithm. Let $\overline{\boldsymbol{qc}}(C_0, \mathcal{R})$ be the minimal set such that:

- $\boldsymbol{qc}(C_0, \mathcal{R}) \subseteq \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$,
- if $\forall Q.\mathfrak{C} \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$, then $\mathfrak{C} \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$,
- if $\exists P.\mathfrak{C} \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$, then $\mathfrak{C} \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$.

Unlike $\boldsymbol{qc}(C_0, \mathcal{R})$, the set $\overline{\boldsymbol{qc}}(C_0, \mathcal{R})$ contains sub-quasi-concepts.

Given an $\mathcal{ALCI}r$ concept C_0 and an RBox $\mathcal{R}$, a *completion tree for* C_0 and $\mathcal{R}$ is a structure of the form $\boldsymbol{T} = (V, E, \ell, \mathfrak{g}, \mathfrak{i}, \mathfrak{l})$, where

- (V, E) is a directed tree;
- for each $(x, y) \in E$, we have $\mathfrak{l}(x, y) \in \mathrm{role}(C_0, \mathcal{R})$; if $\mathfrak{l}(x, y) = R$, then y is called an *R-successor of x*; y is called an *R-neighbour* of x if y is an R-successor of x or x is an $\mathrm{inv}(R)$-successor of y;
- $\ell(x) \subseteq \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$, $\mathfrak{g}(x) \subseteq \bigcup_{i=1}^{l} \boldsymbol{qc}^*(\boldsymbol{r}_i)$ and $\mathfrak{i}(x) \subseteq \{1, \ldots, l\}$, for $x \in V$.

We say that a completion tree $\boldsymbol{T}$ *contains a clash* if there is $x \in V$ such that at least one of the following conditions holds:

- $\bot \in \ell(x)$,
- $\{A, \neg A\} \subseteq \ell(x)$, for a concept name A,
- $i \in \mathfrak{i}(x)$ and there exists $\mathfrak{C} \in (\boldsymbol{qc}^*(\boldsymbol{r}_i) \cap \ell(x)) \smallsetminus \mathfrak{g}(x)$,
- $\mathfrak{C} = (t^r, t^\forall, t^-) \in \ell(x)$ and $\ell(x)|^{\forall}_{\mathrm{inv}(t^r)} \not\subseteq t^-$.

A completion tree that does not contain a clash is called *clash-free*.

To ensure that the tableau algorithm eventually comes to a stop, we use a blocking technique that is similar to the one used in [**13**]. A node $x \in V$ is called *blocked* if it is either directly or indirectly blocked. A node $x \in V$ is *directly blocked* if none of its (not necessarily immediate) E-ancestors is blocked, and there are nodes x', y and y' such that:

- y' is not a root,
- $(x', x) \in E$, $(y', y) \in E$ and y is an E-ancestor of x',
- $\ell(x) = \ell(y)$, $\ell(x') = \ell(y')$ and $\mathfrak{l}(x', x) = \mathfrak{l}(y', y)$.

In this case we say that *y blocks x*. A node y is *indirectly blocked* if one of its E-ancestors is blocked. (Note that the blocking condition is the same as for $\mathcal{SR}^+\mathcal{OIQ}$.)

Our tableau algorithm is nondeterministic. It takes an $\mathcal{ALCI}r$ concept C_0 and an RBox $\mathcal{R}$ as input and returns 'yes' or 'no' to indicate whether C_0 is satisfiable w.r.t. $\mathcal{R}$ or not. It starts by constructing the completion tree $\boldsymbol{T} = (V, E, \ell, \mathfrak{g}, \mathfrak{i}, \mathfrak{l})$, where $V = \{x_0\}$, $E = \emptyset$, $\ell(x_0) = \{C_0, \top\}$, $\mathfrak{g}(x_0) = \emptyset$, $\mathfrak{i}(x_0) = \emptyset$, $\mathfrak{l} = \emptyset$. Then the algorithm non-deterministically applies one of the completion rules given in

TABLE 1. Completion rules for the $\mathcal{ALCIr}$ tableau algorithm.

$(\sqcap)$	if $C_1 \sqcap C_2 \in \ell(x)$, x is not indirectly blocked and $\{C_1, C_2\} \not\subseteq \ell(x)$, then $\ell(x) := \ell(x) \cup \{C_1, C_2\}$
$(\sqcup)$	if $C_1 \sqcup C_2 \in \ell(x)$, x is not indirectly blocked and $\{C_1, C_2\} \cap \ell(x) = \emptyset$, then $\ell(x) := \ell(x) \cup \{D\}$, for some $D \in \{C_1, C_2\}$
$(\exists_c)$	if $\exists S.C \in \ell(x) \cap \mathrm{con}(C_0)$, x is not blocked and x has no S-neighbour y with $C \in \ell(y)$, then create a new node $y \in V$ with $\mathfrak{l}(x, y) := \{S\}$, $\ell(y) := \{C, \top\}$
$(\forall)$	if $\forall R.C \in \ell(x)$, x is not indirectly blocked and there is an R-neighbour y of x such that $C \notin \ell(y)$, then set $\ell(y) := \ell(y) \cup \{C\}$
$(\mathrm{guess}_{\boldsymbol{r}_i})$	if $i \notin \mathfrak{i}(x)$ and x is not indirectly blocked, then 1) non-deterministic guess k and $\{\mathfrak{C}_1, \dots, \mathfrak{C}_k\} \subseteq \boldsymbol{qc}^*(\boldsymbol{r}_i)$, 2) set $\mathfrak{i}(x) := \mathfrak{i}(x) \cup \{i\}$, $\ell(x) := \ell(x) \cup \{\mathfrak{C}_1, \dots, \mathfrak{C}_k\}$, and 3) $\mathfrak{g}(x) := \mathfrak{g}(x) \cup \ell(x) \cap \boldsymbol{qc}^*(\boldsymbol{r}_i)$
$(\mathrm{RI}_{\boldsymbol{r}_i})$	if $i \in \mathfrak{i}(x)$, $\forall R_i.\mathfrak{C} \notin \ell(x)$, $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \ell(x))$ and x is not indirectly blocked, then $\ell(x) := \ell(x) \cup \{\forall R_i.\mathfrak{C}\}$
$(\exists_q)$	if $\exists P.\mathfrak{C} \in \ell(x)$, for $\mathfrak{C} = (t^r, t^\forall, t^-)$, x is not blocked and x has no P-successor y with $\mathfrak{C} \in \ell(y)$, then create a new node $y \in V$ and set $\mathfrak{l}(x, y) := \{P\}$, $\ell(y) := \{\top, \mathfrak{C}\}$
(qc)	if $\mathfrak{C} \in \ell(x)$, for $\mathfrak{C} = (t^r, t^\forall, t^-)$, x is not indirectly blocked and $t^\forall \not\subseteq \ell(x)$, then set $\ell(x) := \ell(x) \cup t^\forall$

Table 1; it keeps doing so till either the current completion graph contains a clash, in which case the answer is 'no', or none of the rules is applicable, in which case the algorithm returns 'yes'.

As a consequence of [**13**], we obtain:

LEMMA 4.2. *The tableau algorithm always terminates.*

LEMMA 4.3. *The tableau algorithm returns 'yes' iff there is a tableau for C_0 w.r.t. $\mathcal{R}$.*

REMARK 4.1. The main difference between the tableau rules in this paper and those in [**13**] is in the rules $(\exists_q)$ and $(\mathrm{guess}_{\boldsymbol{r}_i})$. The rule $(\exists_q)$ creates a new node in the case when a certain node has no *P-successor*, while in [**13**] a new node was created if there was no *P-neighbour*. This simplification is possible because $\mathcal{ALCIr}$

does not have number restrictions. The analogue of the rule (guess$_{\boldsymbol{r}_i}$) in [**13**], for every $D \in \boldsymbol{qc}^*(\boldsymbol{r}_i)$, put either D or $\neg D$ to $\ell(x)$. Here we only guess a polynomial subset $\mathfrak{g}(x)$ of $\boldsymbol{qc}^*(\boldsymbol{r}_i)$ and assume that the remaining (possibly exponentially many) quasi-concepts from $\boldsymbol{qc}^*(\boldsymbol{r}_i)$ are taken with negations.

In the next two sections, we are going to show that if there is a model of $\mathcal{R}$ satisfying C_0, then there exists a tableau for C_0 w.r.t. $\mathcal{R}$ of both *polynomial width* and *polynomial depth*. These results will be used later to show that to check whether such tableaux exist only the polynomial space is required.

4.4. Polynomial width. Observe first that the set $\mathrm{con}(C_0)$ we use for labelling the nodes is of size $l_1 = O(|C_0|)$. Note also that the rule $\exists_q$ always creates a new node if there is no suitable P-successor, even though a suitable P-predecessor may exist(cf. the tableau rules in [**13**]). As a results, the completion tree grows in depth rather than in width. More precisely, we have the following:

LEMMA 4.4. *When applied to C_0 and $\mathcal{R}$, the tableau algorithm generates at most $(l+1) \times l_1$ successors of any node in the completion tree.*

PROOF. We say that a node x (a node y) in a completion tree is an $\mathrm{inv}(Q_j)$-*implicit neighbour* (respectively, a Q_j-*implicit neighbour*) of a node y (respectively, x) if (i) there is no arc Q_j connecting x and y, (ii) there is a predecessor z of y such that x is an $\mathrm{inv}(R_j)$-neighbour or an $\mathrm{inv}(R_j)$-implicit neighbour z, and (iii) y has been created by $(\exists_q)$ applied to z and a quasi-concept $\exists\,\mathrm{inv}(P_j).(t^r, t^\forall, t^-)$, where $t^r = Q_j$. (Because of the rule $(\exists_q)$, in (ii) we do not consider P_j-successors of y.)

Successors to a given node can be generated by the rules $(\exists_c)$ or $(\exists_q)$. The rule $(\exists_c)$ generates at most $|\ell(x) \cap \mathrm{con}(C_0)| < l_1$ successors. On the other hand, a quasi-concept of the form $\exists\,\mathrm{inv}(P_i).\mathfrak{C}$ can be added to $\ell(x)$ by the rule $(\forall)$ when x has an $\mathrm{inv}(R_i)$-neighbour. Such a concept can also be added by the rule (qc) if $R_i = Q_j$, for some j, and x has an $\mathrm{inv}(Q_j)$-implicit neighbours or by the rule (guess$_{\boldsymbol{r}_j}$) if $R_i = \mathrm{inv}(Q_j)$, for some j, and x has an Q_j-implicit neighbours. In the latter case (since $\mathrm{rn}(R_i) = \mathrm{rn}(Q_j)$), we have $\boldsymbol{r}_i \lhd \boldsymbol{r}_j$. Thus, it suffices to compute the number of $\mathrm{inv}(R_i)$-neighbours (including implicit ones) of a given node. By induction on k, we now prove the following property: if $\boldsymbol{r}_{i_k} \lhd \boldsymbol{r}_{i_{k-1}} \lhd \cdots \lhd \boldsymbol{r}_{i_1}$ is a longest chain of dependent RIs beginning with $\boldsymbol{r}_{i_k}$, then any node in the completion tree can have at most $k \times l_1$-many $\mathrm{inv}(R_{i_k})$-neighbours (including implicit ones).

For $k = 1$, all $\mathrm{inv}(R_{i_k})$-successors are created by the rule $(\exists_c)$, and so there are at most l_1-many $\mathrm{inv}(R_{i_k})$-neighbours. For $k > 1$, we have at most l_1-many $\mathrm{inv}(R_{i_k})$-successors created by the rule $(\exists_c)$ and at most $(k-1) \times l_1$-many implicit $\mathrm{inv}(R_{i_k})$-neighbours, where $\mathrm{rn}(Q_{i_{k-1}}) = \mathrm{rn}(R_{i_k})$. In total, we have at most $k \times l_1$-many $\mathrm{inv}(R_{i_k})$-neighbours. Namely, a node x can have implicit $\mathrm{inv}(Q_{i_{k-1}})$-neighbours or $Q_{i_{k-1}}$-neighbours if it has a $P_{i_{k-1}}$-predecessor y with $\mathrm{inv}(R_{i_{k-1}})$-neighbours. By IH, the node y can have at most $(k-1) \times l_1$-many $\mathrm{inv}(R_{i_{k-1}})$-neighbours, and so x can have at most $(k-1) \times l_1$ implicit $\mathrm{inv}(R_{i_k})$-neighbours.

Since $k \leqslant l$, we conclude that the rule $(\exists_q)$ can create at most $l \times l_1$ successors of a given node. These neighbours are not affected by the addition of new quasi-concepts of the form $\exists\,\mathrm{inv}(P_i).\mathfrak{C}$ because $P_i \neq R_j$. Thus, any node can have at most $(l+1) \times l_1$ successors. $\square$

4.5. Polynomial depth. Now we prove that if C_0 is satisfiable w.r.t. $\mathcal{R}$, then it has a finite tableau of both polynomial width and polynomial depth. We do that in three steps.

First, by Lemma 4.4, there is a completion tree whose branching factor is at most $(l+1) \times l_1$. Using this tree, we can construct in the same way as in [**13**] a possibly infinite tableau $\boldsymbol{T} = (\boldsymbol{S}, \ell, \mathcal{E})$, which is a tree with branching factor $\leqslant (l+1) \times l_1$. Then, by Theorem 4.1, we can add some edges to $\boldsymbol{T}$ and construct a concept tableau $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}')$, where $\mathcal{E}' = \mathcal{E}_1 \cup \mathcal{E}_2$, $\mathcal{E}_1 = \mathcal{E}$ and $\mathcal{E}_2 = \mathcal{E}' \smallsetminus \mathcal{E}$. As the basis of the concept tableau is a tree, we call $\boldsymbol{T}'$ a *quasi-tree concept tableau* (QTCT, for short) and write $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}_1, \mathcal{E}_2)$ instead of $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}')$.

Second, we use the QTCT $\boldsymbol{T}'$ to construct a QTCT $\overline{\boldsymbol{T}}$ of polynomial 'depth'. We describe this step in more detail. Suppose u_0 is a root of $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}_1, \mathcal{E}_2)$. We then say that u_0 is of *depth* 0 in $\boldsymbol{T}'$. Now, a node $u \neq u_0$ is of *depth* k in $\boldsymbol{T}'$ if there is a neighbour v of u of depth $k-1$ and there is no neighbour of u of depth $< k-1$.

For a concept C in NNF, we define its *quantifier depth* $d(C)$ by taking

$$\begin{aligned} &d(\bot) = d(\top) = d(A) = d(\neg A) = 0, \quad \text{for } A \in \mathcal{N}_C;\\ &d(C_1 \sqcap C_2) = d(C_1 \sqcup C_2) = \max\{d(C_1), d(C_2)\};\\ &d(\exists R.C) = d(\forall R.C) = d(C) + 1. \end{aligned}$$

A set $\mathcal{P}$ is called *good* w.r.t. $\boldsymbol{T}'$ if either $\mathcal{P} = \{(C_0, u_0)\}$ or $\mathcal{P} = \mathcal{P}' \cup \{(C, u)\}$ where $\mathcal{P}'$ is good w.r.t. $\boldsymbol{T}'$, $C \in \mathfrak{c}(u)$ and one of the following conditions holds:

- there is a concept C' of the form $C_1 \sqcap C_2$ or $C_1 \sqcup C_2$, where $C \in \{C_1, C_2\}$, such that $(C', u) \in \mathcal{P}'$
- there exist an inv(R)-neighbour v of u and a concept C' of the form $\exists R.C$ or $\forall R.C$ such that $(C', v) \in \mathcal{P}'$.

In either case, we say that the concept C *was added to* $\mathcal{P}$ *because of* C'. Clearly, if both $\mathcal{P}$ and $\mathcal{P}'$ are good w.r.t. $\boldsymbol{T}'$, then $\mathcal{P} \cup \mathcal{P}'$ is also good w.r.t. $\boldsymbol{T}'$.

Suppose $\mathcal{P}$ is good w.r.t. $\boldsymbol{T}'$ and $u \in \boldsymbol{S}$. Set $d(u, \mathcal{P}) = \max\{d(C) \mid (C, u) \in \mathcal{P}\}$ if there is a concept C such that $(C, u) \in \mathcal{P}$; otherwise set $d(u, \mathcal{P}) = 0$.

LEMMA 4.5. *If $\mathcal{P} = \mathcal{P}' \cup \{(C, u)\}$ is good w.r.t. $\boldsymbol{T}'$ and C was added to $\mathcal{P}$ because of C', then $d(C') \geqslant d(C)$.*

PROOF. If $C' \in \mathfrak{c}(u)$ is of the form $C_1 \sqcap C_2$ or $C_1 \sqcup C_2$, where $C \in \{C_1, C_2\}$, then $d(C') = \max\{d(C_1), d(C_2)\} \geqslant d(C)$.

If $C' \in \mathfrak{c}(v)$ has the form $\forall R.C$ or $\exists R.C$, where v is an inv(R)-neighbour of u, then $d(C') = d(C) + 1 > d(C)$. □

LEMMA 4.6. *Suppose $\mathcal{P}$ is good w.r.t. $\boldsymbol{T}'$ and u is of depth k in $\boldsymbol{T}'$. If $k \leqslant d(C_0)$, then $d(u, \mathcal{P}) \leqslant d(C_0) - k$.*

PROOF. The proof proceeds by induction on $|\mathcal{P}|$. If $\mathcal{P} = \{(C_0, u_0)\}$, then the claim holds since, for $u \neq u_0$, we have $d(u, \mathcal{P}) = 0$ and $d(u_0, \mathcal{P}) = d(C_0)$. Let $\mathcal{P} = \mathcal{P}' \cup \{(C, v)\}$. By IH, $d(u, \mathcal{P}') \leqslant d(C_0) - k$. If $v \neq u$, then $d(u, \mathcal{P}) = d(u, \mathcal{P}')$ and again the claim holds. If $u = v$ and $d(v, \mathcal{P}) = d(v, \mathcal{P}')$, then the claim is obvious.

If $u = v$ and $d(v,\mathcal{P}) = d(C) > d(v,\mathcal{P}')$, then C has been added to $\mathcal{P}$ because of a concept $C' \in \mathfrak{c}(u')$ of the form $\forall R.C$ or $\exists R.C$, where u' is an inv(R)-neighbour of v. We have $d(C') = d(C)+1 > d(C)$ and u' is of depth k' in $\boldsymbol{T}'$ with $k-1 \leqslant k' \leqslant k+1$. So $d(u,\mathcal{P}) = d(C) = d(C') - 1 \leqslant d(u',\mathcal{P}') - 1 \leqslant d(C_0) - k' - 1 \leqslant d(C_0) - k$. $\square$

Corollary 4.1. *Suppose $\mathcal{P}$ is good w.r.t. $\boldsymbol{T}'$.*
(a) *If a node u is of depth $k = d(C_0)$ in $\boldsymbol{T}'$, then $d(u,\mathcal{P}) = 0$.*
(b) *If u is of depth $k > d(C_0)$ in $\boldsymbol{T}'$, then there is no concept C with $(C,u) \in \mathcal{P}$.*

Let $\overline{\mathcal{P}}$ be a maximal set that is good w.r.t. $\boldsymbol{T}'$, that is,

$$\overline{\mathcal{P}} = \bigcup_{\mathcal{P} \text{ is good w.r.t. } \boldsymbol{T}'} \mathcal{P}.$$

Using the QTCT $\boldsymbol{T}' = (\boldsymbol{S}', \mathfrak{c}, \mathcal{E}_1, \mathcal{E}_2)$ and $\overline{\mathcal{P}}$, we construct $\overline{\boldsymbol{T}} = (\overline{\boldsymbol{S}}, \overline{\mathfrak{c}}, \overline{\mathcal{E}_1}, \overline{\mathcal{E}_2})$ as follows. Let $\boldsymbol{S}_0 = \{u \in S' \mid \text{there is a concept } C \text{ such that } (C,u) \in \overline{\mathcal{P}}\}$ and $\overline{\boldsymbol{S}}$ be a minimal set such that

- $\boldsymbol{S}_0 \subseteq \overline{\boldsymbol{S}} \subseteq \boldsymbol{S}'$,
- if $(u,v) \in \overline{\mathcal{E}}(R_i)$, then $(u,v) \in \overline{\mathcal{E}}(Q_i) \circ \overline{\mathcal{E}}(P_i)$,

where, for any role $R \in \text{role}(C_0,\mathcal{R})$ and $i = 1,2$, we set $\overline{\mathcal{E}_i}(R) = \mathcal{E}_i(R) \cap (\overline{\boldsymbol{S}} \times \overline{\boldsymbol{S}})$ and $\overline{\mathcal{E}}(R) = \overline{\mathcal{E}_1}(R) \cup \overline{\mathcal{E}_2}(R)$. Finally, we set $\overline{\mathfrak{c}}(u) = \{C \mid (C,u) \in \overline{\mathcal{P}}\} \cup \{\top\}$ for $u \in \overline{\boldsymbol{S}}$.

Theorem 4.2. *The structure $\overline{\boldsymbol{T}} = (\overline{\boldsymbol{S}}, \overline{\mathfrak{c}}, \overline{\mathcal{E}_1}, \overline{\mathcal{E}_2})$ is a QTCT with the following properties:*

- *Any $u \in \overline{\boldsymbol{S}}$ is of depth $\leqslant d(C_0) + l$ in $\overline{\boldsymbol{T}}$.*
- *Any path from the root node u_0 to a node u (without repetitions) that uses only $\overline{\mathcal{E}_1}$-arcs is of the length $\leqslant (d(C_0)+l) \times (l+1)$.*
- *Any node $u \in \overline{\boldsymbol{S}}$ has at most $(l+1) \times l_1$-many $\overline{\mathcal{E}_1}$-successors.*

Proof. First, we prove that $\overline{\boldsymbol{T}}$ is a concept tableau.

p1: $C_0 \in \overline{\mathfrak{c}}(u_0)$ since $(C_0,u_0) \in \overline{\mathcal{P}}$;
p2: holds because $\overline{\mathfrak{c}}(u) \subseteq \mathfrak{c}(u)$;
p3: $\top \in \overline{\mathfrak{c}}(u)$ by the definition of $\overline{\mathfrak{c}}(u)$, and $\bot \notin \overline{\mathfrak{c}}(u)$ because $\overline{\mathfrak{c}}(u) \subseteq \mathfrak{c}(u)$;
p4: if $(C_1 \sqcap C_2) \in \overline{\mathfrak{c}}(u) \subseteq \mathfrak{c}(u)$, then $(C_1 \sqcap C_2, u) \in \overline{\mathcal{P}}$, $C_1 \in \mathfrak{c}(u)$ and $C_2 \in \mathfrak{c}(u)$, so $\overline{\mathcal{P}} \cup \{(C_i,u)\}$ is good w.r.t. $\boldsymbol{T}'$ for $i = 1,2$ i.e., $(C_1,u),(C_2,u) \in \overline{\mathcal{P}}$; therefore, $C_1 \in \overline{\mathfrak{c}}(u)$ and $C_2 \in \overline{\mathfrak{c}}(u)$;
p5: is similar to (p4);
p6: if $\exists R.C \in \overline{\mathfrak{c}}(u) \subseteq \mathfrak{c}(u)$, then $(\exists R.C,u) \in \overline{\mathcal{P}}$ and there is some $v \in \boldsymbol{S}$ with $(u,v) \in \mathcal{E}(R)$ and $C \in \mathfrak{c}(v)$, so $\overline{\mathcal{P}} \cup \{(C,v)\}$ is good w.r.t. $\boldsymbol{T}'$, i.e., $(C,v) \in \overline{\mathcal{P}}$; therefore, $v \in \overline{\boldsymbol{S}}$, $C \in \overline{\mathfrak{c}}(v)$ and $(u,v) \in \overline{\mathcal{E}}(R)$;
p7: if $(u,v) \in \overline{\mathcal{E}}(R)$, then $(u,v) \in \mathcal{E}(R)$ and $u,v \in \overline{\boldsymbol{S}}$; so $(v,u) \in \mathcal{E}(\text{inv}(R))$ and $(v,u) \in \overline{\mathcal{E}}(\text{inv}(R))$;
p8: is similar to (p6);
b: if $(u,v) \in \overline{\mathcal{E}}(R_i)$, then $(u,v) \in \overline{\mathcal{E}}(Q_i) \circ \overline{\mathcal{E}}(P_i)$ by the definition of $\overline{\boldsymbol{S}}$ and $\overline{\mathcal{E}}(R)$.

If $u \in \boldsymbol{S}_0$, then, by Lemma 4.6, u is of depth $\leqslant d(C_0)$ in $\overline{\boldsymbol{T}}$. Consider two nodes $u,v \in \overline{\boldsymbol{S}}$ whose depth is $\leqslant k$ in $\overline{\boldsymbol{T}}$. If $(u,v) \in \overline{\mathcal{E}}(R_i)$, then $(u,v) \in \overline{\mathcal{E}}(Q_i) \circ \overline{\mathcal{E}}(P_i)$, and so there is $z \in \overline{\boldsymbol{S}}$ such that $(u, z) \in \overline{\mathcal{E}}(Q_i)$. Clearly, the depth of z in $\overline{\boldsymbol{T}}$ is $\leqslant (k+1)$. The arc $(u, z) \in \overline{\mathcal{E}}'(Q_i)$ can require a new node, but only if $\mathrm{rn}(Q_i) = \mathrm{rn}(R_j)$. But then we have $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$. This means that we can repeat the same procedure at most l times, and so the depth of the new nodes we consider is $\leqslant (k+l)$. Therefore, the depth of $u \in \overline{\boldsymbol{S}}$ in $\overline{\boldsymbol{T}}$ is $\leqslant d(C_0) + l$.

Now, consider nodes $u, v \in \overline{\boldsymbol{S}}$ whose depth in $\overline{\boldsymbol{T}}$ is k and $k+1$, respectively. A path from u to v that uses only $\overline{\mathcal{E}_1}$-arcs is of length $\leqslant (l+1)$. Therefore, a path from the root u_0 to any node u that uses only $\overline{\mathcal{E}_1}$-arcs is of the length $\leqslant (d(C_0)+l) \times (l+1)$.

The property that all nodes $u \in \overline{\boldsymbol{S}}$ have at most $(l+1) \times l_1$ $\overline{\mathcal{E}_1}$-successors follows from the fact that this property holds for QTCT $\boldsymbol{T}'$. $\square$

Finally, starting from QTCT $\overline{\boldsymbol{T}}$, we construct the required tableau by adding appropriate quasi-concepts as described in the proof of the Theorem 4.1. If we omit the arcs from the tableau that belong to $\overline{\mathcal{E}_2}$, then we obtain a tree-shaped tableau $\overline{\boldsymbol{T}}'$. The depth of the tree is bounded by $(d(C_0) + l) \times (l + 1)$ and the branching factor is bounded by $(l + 1) \times l_1$.

By the proof of Lemma 4.4, we conclude that $\ell(x)$ contains at most $l \times l_1$ quasi-concepts of the form $\exists P.\mathfrak{C}$ and there are l quasi-concepts of the form $\forall R_i.\mathfrak{C}$, that is, there are at most $l \times (l_1 + 1)$ quasi-concepts in $\ell(x)$. On the other hand, there are at most l_1 concepts in $\ell(x)$. Therefore, $|\ell(x)| \leqslant l_1 + l \times (l_1 + 1)$.

4.6. PSpace-completeness. PSPACE-hardness of concept satisfiability w.r.t. RBoxes in $\mathcal{ALCIr}^-$ follows from the fact that concept satisfiability in $\mathcal{ALC}$ is PSPACE-complete. To prove a matching upper bound, we give a nondeterministic algorithm that only requires the polynomial space. The algorithm proceeds in 8 steps:

1. Create a node u_0, set $u := u_0$ and guess a set $\ell(u_0)$. If $C_0 \in \ell(u_0)$, then go to step 2; otherwise return 'no'.
2. If the conditions (p2), (p3), (p4), (p5) and (p10) hold for $\ell(u)$, then go to step 3; otherwise return 'no'.
3. Set $\mathfrak{m}(u) := \{\exists R.C \mid \exists R.C \in \ell(u)\} \cup \{\exists P.\mathfrak{C} \mid \exists P.\mathfrak{C} \in \ell(u)\}$. Guess a non-negative integer $k(u)$ (the number of successors) and go to step 4.
4. If $k(u) > 0$, then go to step 5; otherwise go to step 6.
5. Create a new node v; set $k(u) := k(u)-1$. Guess $\ell(v)$ and $R \in \mathrm{role}(C_0, \mathcal{R})$. Set $\mathcal{E}(R) := \mathcal{E}(R) \cup \{(u, v)\}$ and $\mathcal{E}(\mathrm{inv}(R)) := \mathcal{E}(\mathrm{inv}(R)) \cup \{(v, u)\}$. For all $\exists R.C \in \mathfrak{m}(u)$, if $C \in \ell(v)$ (i.e., (p6) holds), then $\mathfrak{m}(u) := \mathfrak{m}(u) \setminus \{\exists R.C\}$. For all $\exists R.(t^r, t^\forall, t^-) \in \mathfrak{m}(u)$, if $t^\forall \subseteq \ell(v)$ (that is, (p11) holds), then $\mathfrak{m}(u) := \mathfrak{m}(u) \setminus \{\exists R.(t^r, t^\forall, t^-)\}$. If (p8) and (p9) hold for u and v, set $u := v$ and go to step 2; otherwise return 'no'.
6. If $\mathfrak{m}(u) = \emptyset$ go to step 7; otherwise return 'no'.
7. If $u = u_0$ (i.e., u is the root), then return 'yes'; otherwise go to step 8.
8. If v is a parent of u with $(u, v) \in \mathcal{E}(R)$, then set $\mathcal{E}(R) := \mathcal{E}(R) \setminus \{(u, v)\}$ and $\mathcal{E}(\mathrm{inv}(R)) := \mathcal{E}(\mathrm{inv}(R)) \setminus \{(v, u)\}$, and set $u := v$. Go to step 4.

Clearly, the algorithm needs only polynomially space. As NPSPACE=PSPACE [**15**], we obtain the following theorem.

THEOREM 4.3. *Concept satisfiability w.r.t. RBoxes in $\mathcal{ALCI}r^-$ is* PSPACE-*complete.*

References

1. F. Baader, *Restricted role-value-maps in a description logic with existential restrictions and terminological cycles*, in: D. Calvanese, G. De Giacomo, E. Franconi (Eds.), *Proc. 2003 Internat. Workshop on Description Logics (DL2003).*
2. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. F. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*, 2nd ed., Cambridge Univ. Press, 2007.
3. C. Bock, X. Zha, H. Suh, J. Lee, *Ontological product modeling for collaborative design*, Advanced Engineering Informatics **24** (2010), 510–524.
4. R. J. Brachman, J. G. Schmolze, *An overview of the KL-ONE knowledge representation system*, Cognitive Science **9**(2) (1985) 171–216.
5. B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, U. Sattler, *OWL 2: The next step for OWL*, J. Web Semantics **6**(4) (2008), 309–322.
6. A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer, *Alternation*, J. ACM **28**(1) (1981), 114–133.
7. S. Demri, *The complexity of regularity in grammar logics and related modal logics*, J. Log. Comput. **11**(6) (2001), 933–960.
8. I. Horrocks, O. Kutz, U. Sattler, *The irresistible $\mathcal{SRIQ}$*, in: B. Cuenca Grau, I. Horrocks, B. Parsia, P. Patel-Schneider (Eds.), *Proc. OWLED*05 Workshop on OWL, Experiences and Directions*,
9. ______, *The even more irresistible $\mathcal{SROIQ}$*, in: P. Doherty, J. Mylopoulos, C. Welty (eds.), *Principles of Knowledge Representation and Reasoning: Proc. Tenth Internat. Conf. (KR-06)*, 57–67,
10. I. Horrocks, U. Sattler, *Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms* Artif. Intell. **160**(1-2) (2004), 79–104.
11. Y. Kazakov, *An extension of complex role inclusion axioms in the description logic $\mathcal{SROIQ}$.* in: J. Giesl, R. Hähnle (Eds.), *Automated Reasoning, 5th Internat. Joint Conf., IJCAR 2010*, Lect. Notes Comput. Sci., Springer-Verlag, 2010, 472–486,.
12. N. Krdžavac, C. Bock, *Reasoning in manufacturing part-part examples with OWL2*, U.S. National Institute of Standards and Technology, Gaithersburg, U.S.A, 2008.
13. M. Mosurović, N. Krdžavac, H. Graves, M. Zakharyaschev, *A Decidable Extension of SROIQ with Complex Role Chains and Unions*, J. Artif. Intell. Res. (JAIR) **47** (2013), 809–851.
14. A. Rector, *Analysis of propagation along transitive roles: formalisation of the GALEN experience with medical ontologies*, in: I. Horrocks, S. Tessaris (Eds.), *Proc 2002 International Workshop on Description Logics (DL2002).*
15. W. J. Savitch, *Relationships between nondeterministic and deterministic tape complexities*, J. Comp. System Sci. **4** (1970), 177–192.
16. M. Schmidt-Schauß, *Subsumption in KL-ONE is undecidable*, in: H. J. Levesque (Ed.), *Proc. First Internat. Conf. on Principles of Knowledge Representation and Reasoning*, M. Kaufmann, 1989, 421–431.

Faculty of Mathematics and Natural Science
University of Montenegro
Podgorica, Montenegro
milenko@ac.me

(Received 29 09 2013)

Department of Computer Science and Information Systems
Birkbeck, University of London, U.K.
michael@dcs.bbk.ac.uk