Journal of Artificial Intelligence Research 47 (2013) 809-851 Submitted 04/13; published 08/13

A Decidable Extension of $\mathcal{SROIQ}$ with Complex Role Chains and Unions

Milenko Mosurovic MILENKO@AC.ME
Faculty of Natural Sciences and Mathematics,
University of Montenegro, Montenegro.

Nenad Krdzavac NENADKR@TESLA.RCUB.BG.AC.RS
Faculty of Organizational Sciences,
University of Belgrade, Serbia.

Henson Graves HENSON.GRAVES@HOTMAIL.COM
Algos Associates, 2829 West Cantey Street,
Fort Worth, TX 76109 U.S.

Michael Zakharyaschev MICHAEL@DCS.BBK.AC.UK
Department of Computer Science and Information Systems,
Birkbeck, University of London, U.K.

Abstract

We design a decidable extension of the description logic $\mathcal{SROIQ}$ underlying the Web Ontology Language *OWL 2*. The new logic, called $\mathcal{SR}^{+}\mathcal{OIQ}$, supports a controlled use of role axioms whose right-hand side may contain role chains or role unions. We give a tableau algorithm for checking concept satisfiability with respect to $\mathcal{SR}^{+}\mathcal{OIQ}$ ontologies and prove its soundness, completeness and termination.

1. Introduction

The ever growing number and scope of application areas puts constant pressure on the designers of ontology languages. Thus, the first version of the Web Ontology Language *OWL*, which became a formal W3C recommendation in 2004, contained the description logic (DL, for short) $\mathcal{SHOIN}$ that allowed the use of the basic DL $\mathcal{ALC}$ together with inverse and transitive roles, role hierarchies, nominals and unqualified cardinality restrictions. Its second reincarnation *OWL 2*, adopted in 2009, is based on a more powerful formalism, $\mathcal{SROIQ}$, which extends $\mathcal{SHOIN}$ with such features as complex role chains, asymmetric, reflexive and disjoint roles, and qualified cardinality restrictions (Horrocks & Sattler, 2004; Horrocks, Kutz, & Sattler, 2006; Cuenca Grau, Horrocks, Motik, Parsia, Patel-Schneider, & Sattler, 2008).

The addition of role inclusions that involve role chains was motivated by multiple use cases in the life sciences domain which require means to describe 'interactions between locative properties and various kinds of part-whole properties' (Cuenca Grau et al., 2008). For example, the role inclusion axiom

$$\mathsf{hasLocation} \circ \mathsf{isPartOf} \sqsubseteq \mathsf{hasLocation}$$

states that if an object x is located in y, and if y is part of z, then x is also located in z (Rector, 2002). However, having resolved the issue of role chains in the left-hand side of role inclusion axioms, as in the example above, $\mathcal{SROIQ}$ and *OWL 2* fall short of providing means to represent such chains and/or unions of roles on the right-hand side, which are often required for modelling structured objects, in particular, in the emerging area of ontological product modelling and collaborative design (Bock, Zha, Suh, & Lee, 2010).

Consider, for example, the product model of cars by Bock (2004) and Krdzavac and Bock (2008), part of which is shown in the UML-like diagram below:

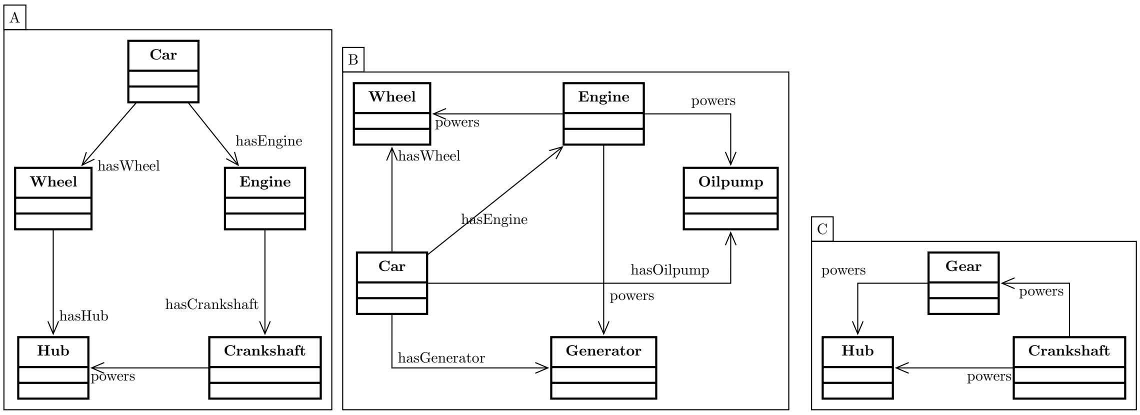


Figure 1: A product model of a car (Krdzavac & Bock, 2008).

The fragment in Fig. 1 (A) involves two statements:

$$\mathsf{hasEngine} \circ \mathsf{hasCrankshaft} \circ \mathsf{powers} \ \sqsubseteq \ \mathsf{hasWheel} \circ \mathsf{hasHub} \tag{1}$$

says that whatever is powered by a crankshaft in an engine of a car is a hub in a wheel of the same car and, conversely,

$$\mathsf{hasWheel} \circ \mathsf{hasHub} \ \sqsubseteq \ \mathsf{hasEngine} \circ \mathsf{hasCrankshaft} \circ \mathsf{powers} \tag{2}$$

states that a hub in a wheel of a car is powered by a crankshaft in an engine of that car. The fragment in Fig. 1 (B) means that an engine in a car can power wheels, the generator and the oil pump, which can be represented by the axiom

$$\mathsf{hasEngine} \circ \mathsf{powers} \ \sqsubseteq \ \mathsf{hasWheel} \sqcup \mathsf{hasGenerator} \sqcup \mathsf{hasOilPump}. \tag{3}$$

Finally, Fig. 1 (C) is supposed to mean that the role powers is transitive:

$$\mathsf{powers} \circ \mathsf{powers} \ \sqsubseteq \ \mathsf{powers}. \tag{4}$$

Role inclusion axioms of the form (1), (2), (4) were a feature of the original KL-ONE terminological language (Brachman & Schmolze, 1985), where they were called 'role-value-maps' and could be applied to certain individuals. Role inclusions with disjunctions on the right-hand side also arise in the context of spatial reasoning with description logics (Wessel, 2001, 2002), where they are used to represent compositions of the $\mathcal{RCC}8$-relations such as $\mathsf{PO} \circ \mathsf{TPP} \subseteq \mathsf{PO} \cup \mathsf{TPP} \cup \mathsf{NTPP}$ (in English: if a region x partially overlaps a region y

and y is a tangential proper part of a region z, then either x partially overlaps z, or x is a tangential proper part of z, or x is a non-tangential proper part of z).

Role inclusions with a complex right-hand side are not allowed by the syntax of $\mathcal{SROIQ}$ and *OWL 2*, which makes adequate representation of models such as in Fig. 1 problematic. Indeed, in these languages, we cannot exclude situations when, for example, *car1* is related to *hub1* via $\mathsf{hasEngine} \circ \mathsf{hasCrankshaft} \circ \mathsf{powers}$ and, at the same time, *hub1* is part of *car2*. Axiom (1) asserts the *existence* of an individual that is a wheel in *car1* and has *hub1*.

The main issue with axioms such as (1) is that they are similar to rewrite rules in semi-Thue systems, the word problem for which is known to be undecidable. One of the simplest examples was given by Tseitin (1956) who showed that the associative calculus (Thue system) with the axioms

$$ac = ca, \quad ad = da, \quad bc = cb, \quad bd = db, \quad edb = be, \quad eca = ae, \quad abac = abacc$$

is undecidable. Schmidt-Schauß (1989) used the undecidability of the word problem to show that the logic underlying KL-ONE is undecidable. Baader (2003) proved (by a reduction of semi-Thue systems) that the tractable description logic $\mathcal{EL}$ becomes undecidable when extended with role inclusions containing role chains on the right-hand side. On the other hand, he observed that role inclusions with a single role on the right-hand side do not increase the complexity of $\mathcal{EL}$. Horrocks and Sattler (2004) proved that the extension of $\mathcal{SHIQ}$ with axioms of the form $R \circ S \sqsubseteq R$ and $S \circ R \sqsubseteq R$ is undecidable; however, decidability can be regained by requiring that such axioms do not involve cycles. Axioms of the form (3) also lead to undecidable logics: Wessel (2001, 2002) showed (by reduction of PCP) that the extension of $\mathcal{ALC}$ with role axioms of the form $S \circ T \sqsubseteq R_1 \sqcup \cdots \sqcup R_n$ is undecidable.

Similar problems have been investigated by the modal logic community. In modal logic, axioms of the form

$$\Box_{i_1} \dots \Box_{i_n} p \to \Box_{j_1} \dots \Box_{j_m} p, \tag{5}$$

known as modal reduction principles, have always attracted attention and still present a great challenge (for example, it is open whether the extension of the basic modal logic K with either of the axioms $\Box\Box\Box p \to \Box\Box p$ or $\Box\Box p \to \Box\Box\Box p$ is decidable). Axioms of the form (5) give rise to grammars generated by the production rules $i_1 \cdot \ldots \cdot i_n \to j_1 \cdot \ldots \cdot j_m$, and the modal logics axiomatised by such axioms are called grammar logics (del Cerro & Penttonen, 1988). It was shown by Demri (2001) and Baldoni (1998) that if this grammar is regular, then the corresponding modal logic is decidable in ExpTime; on the other hand, linear (context-free) grammar logics can be undecidable. It follows, in particular, that the satsifiability problem for $\mathcal{ALC}$ knowledge bases extended with role inclusions $R_1 \dots R_n \sqsubseteq S_1 \dots S_k$ is also ExpTime-complete provided that the grammar generated by the rules $S_1 \dots S_k \to R_1 \dots R_n$ is regular (Demri, 2001, Section 5.3).

In this paper, we design a decidable extension $\mathcal{SR}^{+}\mathcal{OIQ}$ of the description logic $\mathcal{SROIQ}$ that supports a controlled use of role inclusion axioms with a complex right-hand side such as in the examples above. Thus, we can use role inclusion axioms with a chain or union of roles on the right-hand side, and we can also express equality of two role chains or unions such as in (1) and (2). To ensure decidability, we impose certain regularity conditions on the role axioms in a given ontology that generalise the syntactic restrictions of Horrocks et al.

(2006) and Kazakov (2010). These conditions are checked in polynomial time and employed, as a pre-processing step, to build finite automata for some roles in the ontology. Intuitively, the automaton for a role R recognises role chains that are subsumed by R according to the ontology and passes the concept C to the end of the chain whenever its beginning belongs to $\forall R.C$.

Our decision algorithm builds on the tableau technique developed by Horrocks et al. (2006) and uses some ideas of Halpern and Moses (1992, pp. 34-35) in order to pass *sets* of concepts along role chains required by role inclusions with a complex right-hand side such as (1)–(3). If there are no such axioms, our tableau algorithm behaves precisely as the tableau algorithm for $\mathcal{SROIQ}$; otherwise it may suffer multiple exponential blowups (depending on the number of role inclusions with a complex right-hand side).

An alternative approach to modelling complex structures with description logics was suggested by Motik, Cuenca Grau, Horrocks, and Sattler (2009). Their decidable formalism is based on description graphs that can encode axioms of the form (1), but not in the presence of transitivity (4) (in which case the language generated by the role chain in the left-hand side of (1) is infinite and cannot be represented by a finite graph). To ensure decidability, Motik et al. (2009) impose acyclicity conditions on the description graphs and do not allow the same role to appear in the description graph and the DL ontology. For example, we cannot straightforwardly combine a description graph encoding the model in Fig. 1 with a vehicle tax ontology containing axioms such as

$$\mathsf{Car} \sqcap \exists \mathsf{hasEngine.LargeEngine} \sqsubseteq \exists \mathsf{vehicleTax.HigherTax}. \tag{6}$$

In $\mathcal{SR}^+\mathcal{OIQ}$, the addition of (6) to (1)–(4) does not cause a problem.

The structure of the paper is as follows. We define the syntax and semantics of the description logic $\mathcal{SR}^+\mathcal{OIQ}$ in the next two sections. In particular, Section 3 defines and gives the intuition behind the regularity conditions imposed by $\mathcal{SR}^+\mathcal{OIQ}$ on role axioms. The aim of Section 4 is to illustrate by a number of examples the new challenges in the tableau construction we are facing when dealing with $\mathcal{SR}^+\mathcal{OIQ}$ compared to the case of $\mathcal{SROIQ}$. We use these examples to motivate and explain the new ideas, notions and techniques that are required for our tableau-based decision algorithm for $\mathcal{SR}^+\mathcal{OIQ}$. Tableaux for $\mathcal{SR}^+\mathcal{OIQ}$ are defined formally in Appendix A. In Appendix B, we give a tableau algorithm for $\mathcal{SR}^+\mathcal{OIQ}$ and prove that it is sound, complete and always terminates. We discuss the obtained results and open problems in Section 5.

2. Description Logic $\mathcal{SR}^+\mathcal{OIQ}$

We begin by formally defining the syntax and semantics of the description logic $\mathcal{SR}^+\mathcal{OIQ}$. The *alphabet* of $\mathcal{SR}^+\mathcal{OIQ}$ consists of three countably infinite and disjoint sets $\mathcal{N}_C$, $\mathcal{N}_R$ and $\mathcal{N}_I$ of *concept names*, *role names* and *individual names*, respectively. We also distinguish some proper subset $\mathcal{N}_N \subsetneqq \mathcal{N}_C$, whose members are called *nominals*. 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 $\cdot^\mathcal{I}$ is an *interpretation function* that assigns to every $A \in \mathcal{N}_C$ a subset $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, with $A^\mathcal{I}$ being a singleton set if $A \in \mathcal{N}_N$; to every $R \in \mathcal{N}_R$ a binary relation $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$; and to every $a \in \mathcal{N}_I$ an element $a^\mathcal{I} \in \Delta^\mathcal{I}$. Following the *OWL 2* standards, we do not adopt the *unique name assumption* and allow $a^{\mathcal{I}} = b^{\mathcal{I}}$ for distinct $a, b \in \mathcal{N}_I$.

We now introduce the role and concept constructs that are available in $\mathcal{SR}^+\mathcal{OIQ}$. 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 $rn(R) = rn(R^-) = R$, for $R \in \mathcal{N}_R$. We define a *$\mathcal{SR}^+\mathcal{OIQ}$-role* as a *chain* $R_1 \dots R_n$ or a *union* $R_1 \sqcup \dots \sqcup R_n$ of basic roles R_i, and interpret these new constructs by taking

$$\begin{aligned} (R_1 \dots R_n)^{\mathcal{I}} &= R_1^{\mathcal{I}} \circ \dots \circ R_n^{\mathcal{I}}, \\ (R_1 \sqcup \dots \sqcup R_n)^{\mathcal{I}} &= R_1^{\mathcal{I}} \cup \dots \cup R_n^{\mathcal{I}}, \end{aligned}$$

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

In the set $\mathcal{N}_R$ of role names, we distinguish some proper subset $\mathcal{N}_S$ and call its members and their inverses *simple roles*; those basic roles that are not simple will be called *non-simple*. Simple and non-simple roles will have to satisfy different constraints in concepts and role inclusion axioms to be defined below.

$\mathcal{SR}^+\mathcal{OIQ}$-concepts, C, are defined by the following grammar, where $A \in \mathcal{N}_C$, R is a basic role, S a simple role, and n a positive integer (given in binary):

$$\begin{array}{rcl} C & ::= & A \quad | \quad \bot \quad | \quad \top \quad | \quad \neg C \quad | \quad C_1 \sqcap C_2 \quad | \quad C_1 \sqcup C_2 \quad | \\ & & \exists R.C \quad | \quad \forall R.C \quad | \quad \leq nS.C \quad | \quad \geq nS.C \quad | \quad \exists S.\mathit{Self}. \end{array}$$

The interpretation of these concepts is defined as follows, where $\sharp X$ is the cardinality of X:

$$\begin{array}{l} \top^{\mathcal{I}} = \Delta^{\mathcal{I}}, \quad \bot^{\mathcal{I}} = \emptyset, \\ (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, \quad (C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}, \quad (C_1 \sqcup C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}, \\ (\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y \in C^{\mathcal{I}}\, (x, y) \in R^{\mathcal{I}}\}, \\ (\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\, ((x, y) \in R^{\mathcal{I}} \to y \in C^{\mathcal{I}})\}, \\ (\exists S.\mathit{Self})^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid (x, x) \in S^{\mathcal{I}}\}, \\ (\leq n\, S.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid (x, y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leq n\}, \\ (\geq n\, S.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid (x, y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geq n\}. \end{array}$$

A *$\mathcal{SR}^+\mathcal{OIQ}$-knowledge base* (KB, for short) consists of a TBox, an RBox and an ABox. A *TBox*, $\mathcal{T}$, is a finite set of *concept inclusions* (CIs), which are expressions of the form $C_1 \sqsubseteq C_2$. Such a CI is *satisfied* in $\mathcal{I}$ if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$, in which case we write $\mathcal{I} \models C_1 \sqsubseteq C_2$. An *ABox*, $\mathcal{A}$, is a finite set of assertions of the form

$$a : C, \quad (a, b) : R, \quad (a, b) : \neg S, \quad a \neq b,$$

where a and b are individual names, R a basic role, S a simple role, and C a concept. The satisfaction relation for such ABox assertions is given by

$$\begin{aligned}
\mathcal{I} \models a : C \quad &\text{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}}, \\
\mathcal{I} \models (a, b) : R \quad &\text{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}, \\
\mathcal{I} \models (a, b) : \neg S \quad &\text{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \notin S^{\mathcal{I}}, \\
\mathcal{I} \models a \neq b \quad &\text{iff} \quad a^{\mathcal{I}} \neq b^{\mathcal{I}}.
\end{aligned}$$

An *RBox*, $\mathcal{R}$, is a finite set of disjointness constraints and role axioms. A *disjointness constraint* $Dis(S_1, S_2)$ is imposed on *simple* roles S_1, S_2; it is satisfied in $\mathcal{I}$ if $S_1^{\mathcal{I}} \cap S_2^{\mathcal{I}} = \emptyset$. A *role axiom* (RA) can be of the following six types, where S, S' are simple roles; Q', Q, $Q_1, \dots, Q_m$ non-simple roles; and $R, R_1, \dots, R_m$ are arbitrary basic roles:

(A) $S \sqsubseteq S'$, $QQ \sqsubseteq Q$, $Q^- \sqsubseteq Q$,

(B) $R_1 \dots R_m \sqsubseteq Q$, $QR_1 \dots R_m \sqsubseteq Q$, $R_1 \dots R_m Q \sqsubseteq Q$, for $m \geq 1$,

(C) $R \sqsubseteq QR_1 \dots R_m$, for $m \geq 1$,

(D) $R \sqsubseteq Q_1 \sqcup \dots \sqcup Q_m$, for $m > 1$,

(E) $Q' = QR_1 \dots R_m$, for $m \geq 1$,

(F) $Q = Q_1 \sqcup \dots \sqcup Q_m$, for $m > 1$.

RAs of the form (A)–(D) are called *role inclusions* (RIs), while those of the form (E) and (F) *role equalities* (REs). An RBox $\mathcal{R}$ may contain any set of role axioms satisfying the regularity conditions to be defined and discussed in the next section.

Note that, although RAs in $\mathcal{SR}^+\mathcal{OIQ}$ are only restricted to the form (A)–(F), they can encode more general role inclusions of the form (provided that they meet the regularity conditions to be defined below)

$$(R_1^1 \dots R_{n_1}^1) \sqcup \dots \sqcup (R_1^m \dots R_{n_m}^m) \;\sqsubseteq\; (R_1^{m+1} \dots R_{n_{m+1}}^{m+1}) \sqcup \dots \sqcup (R_1^k \dots R_{n_k}^k). \tag{7}$$

(In particular, one can easily write an RBox capturing all the RAs (1)–(4) from the introduction.) A detailed discussion of what can actually be represented by $\mathcal{SR}^+\mathcal{OIQ}$ RBoxes will also be given in the next section.

If ϱ_i is a chain or union of roles, $i = 1, 2$, then $\varrho_1 \sqsubseteq \varrho_2$ (or $\varrho_1 = \varrho_2$) is satisfied in $\mathcal{I}$ if $\varrho_1^{\mathcal{I}} \subseteq \varrho_2^{\mathcal{I}}$ (respectively, $\varrho_1^{\mathcal{I}} = \varrho_2^{\mathcal{I}}$). We say that the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ is *satisfiable* if there exists an interpretation $\mathcal{I}$ satisfying all the members of $\mathcal{T}$, $\mathcal{R}$ and $\mathcal{A}$. In this case we write $\mathcal{I} \models \mathcal{K}$ and call $\mathcal{I}$ a *model* of $\mathcal{K}$.

Our main reasoning problem in this paper is *concept satisfiability with respect to KBs*: given a $\mathcal{SR}^+\mathcal{OIQ}$ concept C and a KB $\mathcal{K}$, decide whether there is a model $\mathcal{I}$ of $\mathcal{K}$ such that $C^{\mathcal{I}} \neq \emptyset$. All other standard reasoning problems such as subsumption, KB satisfiability or instance checking are known to be reducible to concept satisfiability with respect to KBs. Moreover, concept satisfiability with respect to arbitrary KBs can be reduced to concept satisfiability with respect to KBs of the form $(\emptyset, \mathcal{R}, \emptyset)$ (with empty TBoxes and ABoxes); (see Horrocks et al., 2006, Thm. 9).

For a concept C, we denote by $nom(C)$ the set of all nominals that occur in C, and by $role(C)$ the set of all basic roles R such that either R or $inv(R)$ occurs in C; $role(C, \mathcal{K})$ and $role(C, \mathcal{R})$ contain those basic roles and their inverses that occur in C or $\mathcal{K}/\mathcal{R}$.

3. Regular RBoxes

As mentioned in the introduction, unrestricted RAs can easily simulate all kinds of undecidable problems. In this section, we define regular RBoxes that are allowed in $\mathcal{SR}^+\mathcal{OIQ}$. For $\mathcal{SROIQ}$ RAs—that is, RAs of the form (A) and (B)—our restrictions are the same as those used by Kazakov (2010). As suggested by the term 'regular,' we are going to use the regularity restrictions to construct finite automata for roles R that recognise role chains subsumed by R in the RBox in question.

Suppose $\mathcal{R}$ is a set of RAs. To define the regularity conditions (to be given in Definition 3), we require the following binary relation $\prec'$ on the set of role names occur in $\mathcal{R}$:

- $rn(R_i) \prec' rn(Q)$, $i = 1, \dots, m$, for RIs of type (B),
- $rn(R) \prec' rn(Q)$, for RIs of type (C),
- $rn(R) \prec' rn(Q_i)$, $i = 1, \dots, m$, for RIs of type (D),
- $rn(R_i) \prec' rn(Q')$, $i = 1, \dots, m$, for REs of type (E).

Denote by $\preceq_\mathcal{R}$ the transitive and reflexive closure of $\prec'$. We write $R_1 \simeq_\mathcal{R} R_2$ if both $R_1 \preceq_\mathcal{R} R_2$ and $R_2 \preceq_\mathcal{R} R_1$, and $R_1 \prec_\mathcal{R} R_2$ if $R_1 \preceq_\mathcal{R} R_2$ and $R_2 \not\preceq_\mathcal{R} R_1$. By the *depth* $d_\mathcal{R}(R)$ of R in $\mathcal{R}$ we understand the largest n for which there exists a chain $R_1 \prec_\mathcal{R} R_2 \prec_\mathcal{R} \cdots \prec_\mathcal{R} R_n \prec_\mathcal{R} R$.

We represent $\mathcal{R}$ as the union $\mathcal{R} = \mathcal{R}_A \cup \mathcal{R}_B \cup \mathcal{R}_C \cup \mathcal{R}_D \cup \mathcal{R}_E \cup \mathcal{R}_F$, where $\mathcal{R}_X$ contains those RAs from $\mathcal{R}$ that are of the form (X), $X \in \{A, B, C, D, E, F\}$. We also write $\mathcal{R}_{A,B}$ for $\mathcal{R}_A \cup \mathcal{R}_B$, etc.

For an RI $\boldsymbol{r} = (\varrho \sqsubseteq R) \in \mathcal{R}_{A,B}$ and role chains ϱ' and ϱ'', we write $\varrho' \sqsubseteq_{\boldsymbol{r}} \varrho''$ if either $\varrho' = \varrho'_1 \varrho \varrho'_2$ and $\varrho'' = \varrho'_1 R \varrho'_2$, or $\varrho' = \varrho'_1 inv(\varrho) \varrho'_2$ and $\varrho'' = \varrho'_1 inv(R) \varrho'_2$, for some ϱ'_1 and ϱ'_2. We write $\varrho' \sqsubseteq_\mathcal{R} \varrho''$ if $\varrho' \sqsubseteq_{\boldsymbol{r}} \varrho''$, for some $\boldsymbol{r} \in \mathcal{R}_{A,B}$, and denote by $\sqsubseteq^*_\mathcal{R}$ the reflexive and transitive closure of $\sqsubseteq_\mathcal{R}$. It follows immediately from the definitions of $\preceq_\mathcal{R}$ and $\sqsubseteq^*_\mathcal{R}$ that we have:

Lemma 1 *If $\varrho = \varrho' R' \varrho''$ and $\varrho \sqsubseteq^*_\mathcal{R} R$, then $rn(R') \preceq_\mathcal{R} rn(R)$.*

Following Kazakov (2010), we say that an RI $(\varrho \sqsubseteq R') \in \mathcal{R}_{A,B}$ is *stratified* in $\mathcal{R}$ if, for every $R \simeq_\mathcal{R} R'$ with $\varrho = \varrho_1 R \varrho_2$, there exists R_1 such that $\varrho_1 R \sqsubseteq^*_\mathcal{R} R_1$ and $R_1 \varrho_2 \sqsubseteq^*_\mathcal{R} R'$. We call $\mathcal{R}_{A,B}$ *stratified* if every RI $\varrho \sqsubseteq R$ with $\varrho \sqsubseteq^*_\mathcal{R} R$ is stratified in $\mathcal{R}$.

For every role R in $\mathcal{R}$, we define the following language $L_\mathcal{R}(R)$ of role chains regarded as words over basic roles:

$$L_\mathcal{R}(R) \;=\; \{\varrho \mid \varrho \sqsubseteq^*_\mathcal{R} R\}.$$

Theorem 2 (Kazakov, 2010) *Suppose $\mathcal{R}$ is an RBox with stratified $\mathcal{R}_{A,B}$. Then the language $L_\mathcal{R}(R)$ is regular, for every role R in $\mathcal{R}$. Moreover, one can construct a non-deterministic finite automaton recognising $L_\mathcal{R}(R)$ the number of transitions in which does not exceed $O(|\mathcal{R}|^{2d_\mathcal{R}(R)})$.*

We are now in a position to define regular RBoxes.

Definition 3 An RBox $\mathcal{R}$ is called *regular* if the following conditions are satisfied:

(c1) $\mathcal{R}_{A,B}$ is stratified;

(c2) $rn(R) \prec_{\mathcal{R}} rn(Q)$, for RIs of type (C);

(c3) $rn(R) \prec_{\mathcal{R}} rn(Q_i)$, $i = 1, \dots, m$, for RIs of type (D);

(c4) $rn(R_i) \prec_{\mathcal{R}} rn(Q')$, $i = 1, \dots, m$, for RAs of type (E);

(c5) there exists a quasi-order $\preceq^1_{\mathcal{R}} \supseteq \preceq_{\mathcal{R}}$ for which

- $rn(Q') \prec^1_{\mathcal{R}} rn(Q)$, for each RA of type (E);
- $rn(Q) \prec^1_{\mathcal{R}} rn(Q_i)$, $i = 1, \dots, m$, for each RA of type (F);

(c6) there exists a quasi-order $\preceq^2_{\mathcal{R}} \supseteq \preceq_{\mathcal{R}}$ for which

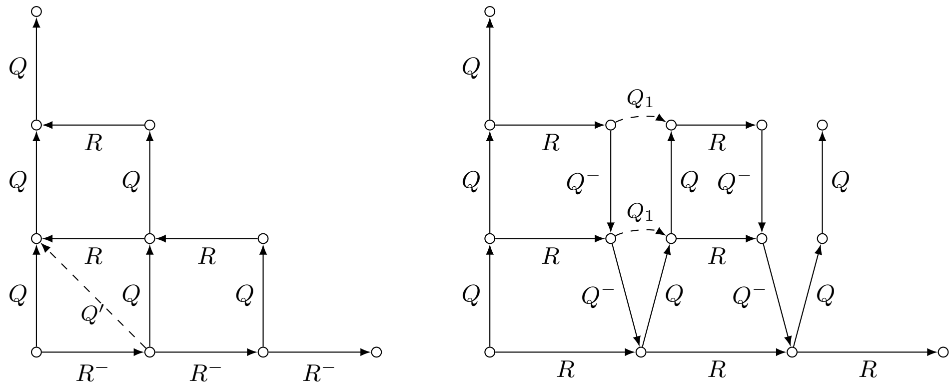
- $rn(Q) \prec^2_{\mathcal{R}} rn(Q')$, for each RA of type (E);
- $rn(Q_i) \prec^2_{\mathcal{R}} rn(Q)$, $i = 1, \dots, m$, for each RA of type (F);

(c7) there do not exist RAs $\boldsymbol{r}$ and $\boldsymbol{r'}$ such that one of the following conditions holds:

- $\boldsymbol{r'} = (Q' = Q_0 R_1 \dots R_{m'})$, $\boldsymbol{r} = (Q = Q_1 \sqcup \dots \sqcup Q_m)$, $rn(Q') = rn(Q)$ and $rn(Q_0) = rn(Q_j)$, for some j, $1 \leq j \leq m$;
- $\boldsymbol{r'} = (Q'_0 = Q_0 R'_1 \dots R'_{m'})$, $\boldsymbol{r} = (Q'_1 = Q_1 R_1 \dots R_m)$, $rn(Q'_0) = rn(Q'_1)$ and $rn(Q_0) = rn(Q_1)$;
- $\boldsymbol{r'} = (Q' = Q'_1 \sqcup \dots \sqcup Q'_{m'})$, $\boldsymbol{r} = (Q = Q_1 \sqcup \dots \sqcup Q_m)$, $rn(Q') = rn(Q)$ and $rn(Q'_i) = rn(Q_j)$, for some i, j, $1 \leq i \leq m'$, $1 \leq j \leq m$.

In the remainder of this section, we discuss the regularity conditions **(c1)**–**(c7)** and illustrate them by concrete examples. Note first that condition **(c1)** is required to ensure decidability of $\mathcal{SROIQ}$; as mentioned in the introduction, dropping it immediately leads to undecidability (Demri, 2001; Horrocks & Sattler, 2004). To understand **(c2)**, consider the following:

Example 4 Let $\mathcal{R} = \{RQ \sqsubseteq Q', Q' \sqsubseteq QR\}$. The former RI is of type (B), while the latter one is of type (C). Clearly, $Q' \sqsubseteq QR$ does not satisfy **(c2)**, and so the RBox is not regular. To see why this situation is 'dangerous,' we observe that $\mathcal{R} \models RQ \sqsubseteq QR$. Now, if the TBox generates infinite chains of Q- and R^--arrows starting from the same point, then the RI $RQ \sqsubseteq QR$ would generate the $\mathbb{N} \times \mathbb{N}$-grid shown on the left-hand side of the picture below:



It is routine then to reduce the undecidable $\mathbb{N} \times \mathbb{N}$-tiling problem to KB satisfiability.

On the other hand, the RBox $\mathcal{R}' = \{R \sqsubseteq QRQ^-\}$ is regular ($rn(R) \prec_{\mathcal{R}'} rn(Q)$). However, it cannot generate a proper $\mathbb{N} \times \mathbb{N}$-grid (as shown on the right-hand side of the picture above). To be able to encode the $\mathbb{N} \times \mathbb{N}$-tiling problem, we require additional RIs such as $Q^-Q \sqsubseteq Q_1$ and $Q^-Q_1Q \sqsubseteq Q_1$. But then the resulting RBox will not satisfy condition **(c1)**.

Condition **(c3)** is similar to **(c2)**; that its omission leads to undecidability was shown by Wessel (2002). To illustrate **(c4)**, we give one more example.

Example 5 Consider the RBox $\mathcal{R} = \{Q'Q \sqsubseteq Q_1,\ Q' = Q^-Q_1\}$. Clearly, it does not satisfy **(c4)**, but with this condition omitted, we only have $Q' \prec_{\mathcal{R}} Q_1$ and $Q \prec_{\mathcal{R}} Q_1$. Now observe that the 'dangerous' RI $Q^-Q_1Q \sqsubseteq Q_1$ from Example 4 is a consequence of $\mathcal{R}$.

Since the REs (E) and (F) imply $Q' \sqsubseteq QR_1 \dots R_m$ and $Q \sqsubseteq Q_1 \sqcup \dots \sqcup Q_m$ of types (C) and (D), condition **(c5)** is similar to **(c2)** and **(c3)**. For **(c6)**, consider the following:

Example 6 The RBox $\mathcal{R} = \{Q_1Q_2 \sqsubseteq Q_2,\ Q_3Q_4 \sqsubseteq Q_3,\ Q_4 = Q_2S\}$ is regular. However, $\mathcal{R}_1 = \mathcal{R} \cup \{Q_1 = Q_3S'\}$ is not regular because $rn(Q_1) \prec_{\mathcal{R}_1} rn(Q_2)$, $rn(Q_4) \prec_{\mathcal{R}_1} rn(Q_3)$, $rn(S) \prec_{\mathcal{R}_1} rn(Q_4)$, $rn(S') \prec_{\mathcal{R}_1} rn(Q_1)$, and so **(c1)**–**(c5)** and **(c7)** are satisfied, while **(c6)** is not. Now, $\mathcal{R}_1$ implies $Q_3S'Q_2 \sqsubseteq Q_2$ and $Q_3Q_2S \sqsubseteq Q_3$, from which we obtain $Q_3Q_2SS'Q_2 \sqsubseteq Q_2$. The RBox containing this RI generates a language that is not regular.

Finally, we require condition **(c7)** in view of the following:

Example 7 The RAs $Q' = QR$ and $Q' = Q \sqcup Q_1$ clearly imply $Q \sqsubseteq QR$. As we saw in Example 4, in the presence of the RI $RQ \sqsubseteq Q$, this would lead to undecidability. Condition **(c7)** does not allow RBoxes of this sort to be counted as regular.

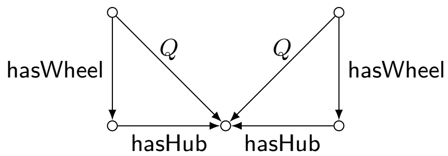
As was already noted, we restrict $\mathcal{SR}^+\mathcal{OIQ}$ RBoxes to RAs of types (A)–(F) mainly in order to simplify notation and proofs; see (7). Every RI $R_1 \dots R_n \sqsubseteq P_1 \dots P_m$ is equivalent to the RI $inv(R_n) \dots inv(R_1) \sqsubseteq inv(P_m) \dots inv(P_1)$. In particular, the RI $inv(R) \sqsubseteq inv(Q_m) \dots inv(Q_1) inv(Q)$ is equivalent to the RI $R \sqsubseteq QQ_1 \dots Q_m$ of type (C), and so we can use the former in $\mathcal{SR}^+\mathcal{OIQ}$ RBoxes provided that $rn(R) \prec rn(Q)$. Every RI $R_1 \dots R_n \sqsubseteq P_1 \dots P_k P'_1 \dots P'_m$ can be replaced with the RIs $R_1 \dots R_n \sqsubseteq P_1 \dots P_k T$ and $T \sqsubseteq P'_1 \dots P'_m$, for a fresh role name T, without affecting the satisfiability of the KB. In particular, if $rn(R_i) \prec rn(Q)$, then we can represent $R_1 \dots R_n \sqsubseteq P_1 \dots P_k Q P_{k+1} \dots P_m$ by means of three $\mathcal{SR}^+\mathcal{OIQ}$ RIs: $R_1 \dots R_n \sqsubseteq R$, $inv(R) \sqsubseteq inv(T) inv(P_k) \dots inv(P_1)$ and $T \sqsubseteq Q P_{k+1} \dots P_m$, for fresh role names R and T. Instead of $R_1 \dots R_n \sqsubseteq P_1 \sqcup \dots \sqcup P_m$ we use $R_1 \dots R_n \sqsubseteq R$ and $R \sqsubseteq P_1 \sqcup \dots \sqcup P_m$, for a fresh role name R. The same can be done for role equality axioms.

The *reflexivity constraint* $Ref(R)$ (saying that $R^{\mathcal{I}}$ is reflexive) can be expressed by means of the RI $S \sqsubseteq R$ and CI $\top \sqsubseteq \exists S.Self$, where S is a fresh simple role.

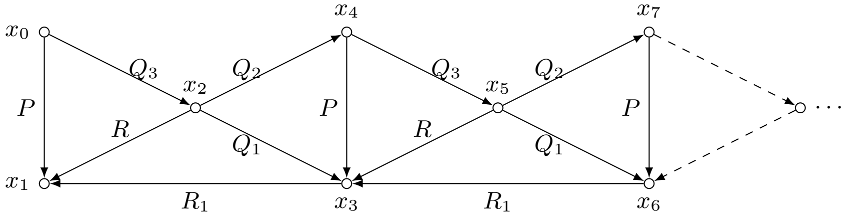
Example 8 The RI (1) from the introduction is represented in $\mathcal{SR}^+\mathcal{OIQ}$ by two RIs:

$$\mathsf{hasEngine} \circ \mathsf{hasCrankshaft} \circ \mathsf{powers} \ \sqsubseteq \ Q, \tag{8}$$
$$Q \ \sqsubseteq \ \mathsf{hasWheel} \circ \mathsf{hasHub}, \tag{9}$$

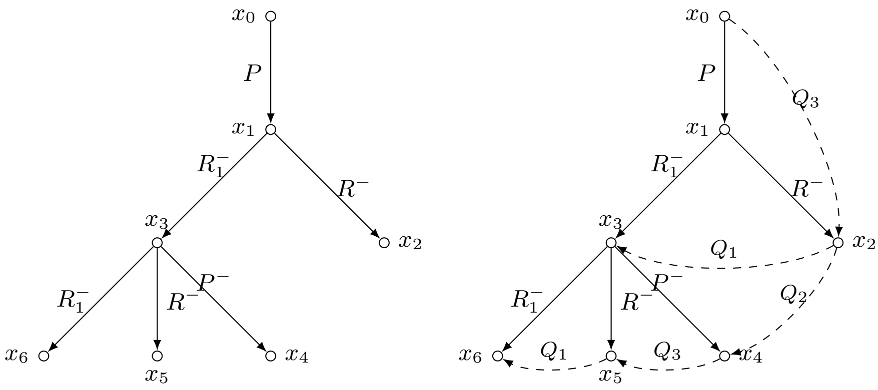
where Q is a fresh non-simple role name. One might suggest that (9) could be replaced with the RI $Q \circ \mathsf{hasHub}^- \sqsubseteq \mathsf{hasWheel}$. However, this is not the case: the interpretation given below satisfies the former but not the latter (obviously, $Q \circ \mathsf{hasHub}^- \sqsubseteq \mathsf{hasWheel}$ does not imply (9)).



Example 9 Consider the (regular) RBox $\mathcal{R} = \{R \sqsubseteq Q_1 R_1,\ Q_1 \sqsubseteq Q_2 P,\ P = Q_3 R\}$ and the ABox $\mathcal{A} = \{(x_0, x_1) : P\}$. Any model of $\mathcal{R}$ and $\mathcal{A}$ contains a sequence of (not necessarily distinct) points $x_0, x_1, x_2, \dots$ arranged according to the patter shown in the picture below:



When applying the tableau algorithm to $\mathcal{R}$ and $\mathcal{A}$ (to be introduced in the remainder of the paper), we construct the same model, but represent it as a tree-shaped structure by omitting the Q_1-, Q_2- and Q_3-arrows, which can always be restored. (In general, we always omit the first role on the right-hand side of an axiom of type (C) and (E), and all roles on the right-hand side of an axiom of type (D) and (F).) This is illustrated in the picture below.



4. $\mathcal{SR}^+\mathcal{OIQ}$ Tableaux by Examples

We prove decidability of $\mathcal{SR}^+\mathcal{OIQ}$ using a tableau-based algorithm, which is a generalisation of the algorithm given by Horrocks et al. (2006). We assume that the reader is familiar with the tableau technique for standard DLs such as $\mathcal{ALCI}$ (Baader, Calvanese, McGuinness, Nardi, & Patel-Schneider, 2003). Our aim in this section is to explain, using concrete examples, both the problems one encounters when constructing tableaux for $\mathcal{SR}^+\mathcal{OIQ}$ and the way to resolve these problems suggested in the paper. Having worked through the examples, the reader will have grasped the general idea of the tableaux for $\mathcal{SR}^+\mathcal{OIQ}$.

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 $con(C)$ the smallest set that contains C and is closed under sub-concepts and $\neg$. For a KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$, we denote by $con(\mathcal{K})$ the union of $con(C)$, for all concepts C occurring in $\mathcal{K}$. For a basic role R and $\Sigma \subseteq con(\mathcal{K})$, we set $\Sigma|_R^\forall = \{C \mid \forall R.C \in \Sigma\}$.

4.1 RIs with Role Chains on the Right-Hand Side

Example 10 Consider first the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$, where

$$\mathcal{A} = \{a : A\}, \quad \mathcal{R} = \{R \sqsubseteq QP\}, \quad \mathcal{T} = \{A \sqsubseteq \exists R.\top,\ A \sqsubseteq \forall Q.B,\ A \sqsubseteq \forall Q.C\}.$$

We start the construction of a tableau for $\mathcal{K}$ by applying the standard tableau rules for $\mathcal{ALC}$. Thus, we create a root node x_0 (corresponding to the only ABox individual a) and label it with $\ell(x_0) = \{A, \exists R.\top, \forall Q.B, \forall Q.C\}$, indicating thereby (some of) the concepts that should contain a according to $\mathcal{K}$. In view of $\exists R.\top \in \ell(x_0)$, we then create an R-successor x_1 of x_0. The interpretation, corresponding to the resulting tableau and shown on the left-hand side of the picture below, is clearly a model of $\mathcal{T}$ and $\mathcal{A}$, but not of $\mathcal{R}$.

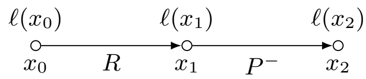
To satisfy $\mathcal{R}$, we need a Q-successor x_2 of x_0, which has x_1 as its P-successor. However, the resulting interpretation, shown on the right-hand side of the picture above, is not a tree. To keep the tableau tree-shaped, we would prefer to create x_2 as a P^--successor of x_1 without drawing the Q-arrow from x_0 to x_2 explicitly. To trigger the creation of x_2 and to ensure that a Q-arrow can always be inserted between x_0 and x_2, we add to each label $\ell(x_i)$ a new 'quasi-concept' of the form $\forall R.\exists P^-.\ell(x_i)|^{\forall}_{Q}$, which encodes $R \sqsubseteq QP$. The intended meaning of this quasi-concept is as expected: every R-successor of x_i must have a P^--successor whose label contains all the concepts in $\ell(x_i)|^{\forall}_{Q} = \{C \mid \forall Q.C \in \ell(x_i)\}$. (Note that tableau nodes are not part of the syntax for quasi-concepts. The quasi-concepts, in fact, extend the syntax with the expressions such as $\exists R.S$, where S is a set of ordinary concepts.) If we agree to extend the standard tableau rules for $\forall R$ and $\exists P^-$ to such quasi-concepts, then we only need one new tableau rule (which will be generalised later on in the paper):

(r1) if $(R \sqsubseteq QP) \in \mathcal{R}$ and $\forall R.\exists P^-.\ell(x)|^{\forall}_{Q} \notin \ell(x)$, then set $\ell(x) := \ell(x) \cup \{\forall R.\exists P^-.\ell(x)|^{\forall}_{Q}\}$.

Now, returning to our example, we apply (r1) to $\ell(x_0)$, $\ell(x_1) = \emptyset$ and obtain:

$$\ell(x_0) := \big\{A,\ \exists R.\top,\ \forall Q.B,\ \forall Q.C,\ \forall R.\exists P^-.\{B, C\}\big\},$$
$$\ell(x_1) := \big\{\forall R.\exists P^-.\emptyset,\ \exists P^-.\{B, C\}\big\}.$$

We then create a P^--successor x_2 of x_1 with $\ell(x_2) = \{B, C, \forall R.\exists P^-.\emptyset\}$, as in the picture below, and stop with a complete and clash-free tableau, which gives a model of $\mathcal{K}$ if we insert the missing Q-arrow between x_0 and x_2.



Note that inserting the missing Q-arrow in the example above becomes more problematic if we extend $\mathcal{T}$ with the CI $B \sqsubseteq \forall Q^-.\neg A$ because then we shall have to add $\neg A$ to $\ell(x_0)$, and obtain a clash. However, we cannot do this without constructing that arrow explicitly.

To cope with this problem, together with $\ell(x_0)|^{\forall}_{Q}$, we can also pass to $\ell(x_2)$ the set $\ell^-_Q(x_0)$ of those concepts $C \in \ell(x_0)$ that can potentially occur in $\forall Q^-.C \in \ell(x_2)$, namely, the set $\ell(x_0) \cap con(\mathcal{K})|^{\forall}_{Q^-}$. We can store this set in some special 'memory' of x_2 in order to compare it with $\ell(x_2)|^{\forall}_{Q^-}$: if $\ell(x_2)|^{\forall}_{Q^-} \not\subseteq \ell^-_Q(x_0)$, then we report a clash. However, this does not solve our problem yet. To see why, consider the extension of $\mathcal{T}$ with $B \sqsubseteq \forall Q^-.C$ (rather than $B \sqsubseteq \forall Q^-.\neg A$). As C does not belong to $\ell(x_0)$, we would have to report a clash, though an addition of C to $\ell(x_0)$ would not lead to a contradiction. A solution we suggest for such situations is to make sure that, for every concept D in $\{C \mid \forall Q^-.C \in con(\mathcal{K})\}$ and $\{\forall Q.C \mid \forall Q.C \in con(\mathcal{K})\}$, either $D \in \ell(x_0)$ or $\neg D \in \ell(x_0)$.

To formalise the idea above as tableau rules, we require some new notation. We allow *quasi-concepts* of the form $\ell_Q(x) = (t^r, t^{\forall}, t^-)$, where $t^r = Q$, $t^{\forall} = \ell(x)|^{\forall}_{Q}$ and $t^- = \ell(x) \cap con(\mathcal{K})|^{\forall}_{Q^-}$; we also denote the first component of this triple by $\ell^r_Q(x)$, the second by $\ell^{\forall}_Q(x)$, and the third by $\ell^-_Q(x)$. The special memory associated with node x will be denoted by $\mathfrak{m}(x)$; we assume that originally it is empty. We require the following tableau rules, which supersede the former (r1):

(r1) if $(R \sqsubseteq QP) \in \mathcal{R}$, rule (r3) is not applicable, and $\forall R.\exists P^-.\ell_Q(x) \notin \ell(x)$, then set $\ell(x) := \ell(x) \cup \{\forall R.\exists P^-.\ell_Q(x)\}$;

(r2) if $\exists P^-.t \in \ell(x)$, for $t = (t^r, t^\forall, t^-)$, and x has no P^--neighbour[1] y with $t^\forall \subseteq \ell(y)$ and $t \in \mathfrak{m}(y)$, then we create a new P^--successor y of x and set $\ell(y) = t^\forall$ and $\mathfrak{m}(y) = \{t\}$;

(r3) if $(R \sqsubseteq QP) \in \mathcal{R}$ and there is $D \in \{\forall Q.C \mid \forall Q.C \in con(\mathcal{K})\} \cup \{C \mid \forall Q^-.C \in con(\mathcal{K})\}$ with $\{D, \neg D\} \cap \ell(x) = \emptyset$, then we set $\ell(x) := \ell(x) \cup \{E\}$, for some $E \in \{D, \neg D\}$;

(clash) if $(t^r, t^\forall, t^-) \in \mathfrak{m}(x)$ and $\ell(x)|^{\forall}_{inv(t^r)} \not\subseteq t^-$, then report a clash.

Example 11 To illustrate, consider the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$, where

$$\mathcal{A} = \{a : A\}, \quad \mathcal{R} = \{R \sqsubseteq QP\}, \quad \mathcal{T} = \{A \sqsubseteq \exists R.\top,\ A \sqsubseteq \forall Q.B,\ B \sqsubseteq \forall Q^-.C,\ C \sqsubseteq \forall Q.D\}.$$

We obtain the following complete and clash-free tableau for $\mathcal{K}$:

$\ell(x_0) = \{A, \exists R.\top, \forall Q.B\},$ (by $\sqsubseteq$)

$\ell(x_0) := \ell(x_0) \cup \{\forall Q.D, C\},$ (by r3)

$\ell(x_0) := \ell(x_0) \cup \big\{\forall R.\exists P^-.\ell_Q(x_0)\big\},\ \ell^\forall_Q(x_0) = \{B, D\},\ \ell^-_Q(x_0) = \{C\},$ (by r1)

create x_1 with $x_0 R x_1$, $\ell(x_1) = \{\forall Q.B, \forall Q.D, \neg C\},$ (by $\exists R$, r3)

$\ell(x_1) := \ell(x_1) \cup \big\{\forall R.\exists P^-.\ell_Q(x_1), \exists P^-.\ell_Q(x_0)\big\},\ \ell^\forall_Q(x_1) = \{B, D\},\ \ell^-_Q(x_1) = \emptyset,$ (by r1, $\forall R$)

create x_2 with $x_1 P^- x_2$, $\mathfrak{m}(x_2) = \{\ell_Q(x_0)\}$ and $\ell(x_2) = \ell^\forall_Q(x_0) = \{B, D\},$ (by r2)

$\ell(x_2) := \ell(x_2) \cup \big\{\forall Q^-.C, \forall Q.B, \forall Q.D, C, \forall R.\exists P^-.\ell_Q(x_2))\big\}.$ (by $\sqsubseteq$, r3, r1)

There is no clash because $\ell(x_2)|^{\forall}_{inv(Q)} \subseteq \ell^-_Q(x_0)$.

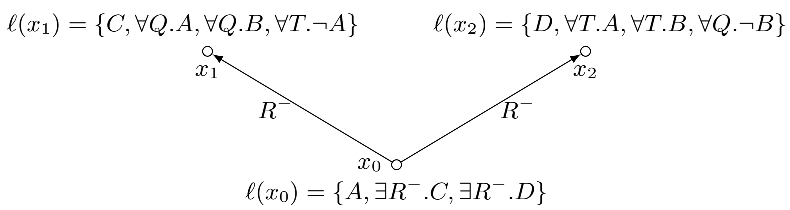
4.2 RIs with Role Unions on the Right-Hand Side

Our next example illustrates tableaux for RIs with unions on the right-hand side.

Example 12 Consider the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ with

$$\mathcal{A} = \{a : A\}, \qquad \mathcal{R} = \{R \sqsubseteq Q \sqcup T\},$$
$$\mathcal{T} = \{A \sqsubseteq \exists R^-.C,\ A \sqsubseteq \exists R^-.D,\ C \sqsubseteq \forall Q.A,\ C \sqsubseteq \forall Q.B,\ C \sqsubseteq \forall T.\neg A,$$
$$D \sqsubseteq \forall T.A,\ D \sqsubseteq \forall T.B,\ D \sqsubseteq \forall Q.\neg B\}.$$

By applying the standard rules, we obtain the tableau shown in the picture below:



1. Intuitively, a neighbour is a successor or a predecessor of a given node. A formal definition of this notion will be given in Section B.

Now, to satisfy $\mathcal{R}$, we have to draw either a Q- or a T-arrow from x_1 to x_0, and also from x_2 to x_0. As before, we do not do this explicitly. To ensure that such arrows can always be drawn, we add to each $\ell(x_i)$ a quasi-concept of the form $\forall R.(\ell(x_i)|_Q^\forall \vee \ell(x_i)|_T^\forall)$, where as before $\ell(x_i)|_P^\forall = \{C \mid \forall P.C \in \ell(x_i)\}$. The meaning of this quasi-concept should be self-evident. Thus, we extend the $\ell(x_i)$ to:

$$\begin{aligned}
\ell(x_0) &:= \ell(x_0) \cup \big\{\forall R.(\emptyset \vee \emptyset)\big\},\\
\ell(x_1) &:= \ell(x_1) \cup \big\{\forall R.(\{A, B\} \vee \{\neg A\})\big\},\\
\ell(x_2) &:= \ell(x_2) \cup \big\{\forall R.(\{\neg B\} \vee \{A, B\})\big\}.
\end{aligned}$$

But then we have to add either A, B or $\neg A$ to $\ell(x_0)$ in view of the quasi-concept in $\ell(x_1)$, and also either $\neg B$ or A, B in view of the quasi-concept in $\ell(x_2)$. The only clash-free way of doing this is to extend $\ell(x_0)$ with A, B. Clearly, we can draw a Q-arrow from x_1 to x_0 and a T-arrow from x_2 to x_0.

We can now formulate tableau rules for handling role unions in RIs, taking into account quasi-concepts with triples considered above:

(r4) if $(R \sqsubseteq Q \sqcup T) \in \mathcal{R}$ and there is $D \in \{\forall P.C \mid \forall P.C \in \mathit{con}(\mathcal{K})\} \cup \{C \mid \forall P^-.C \in \mathit{con}(\mathcal{K})\}$, for $P \in \{Q, T\}$ with $\{D, \neg D\} \cap \ell(x) = \emptyset$, then we set $\ell(x) := \ell(x) \cup \{E\}$, for some $E \in \{D, \neg D\}$;

(r5) if $(R \sqsubseteq Q \sqcup T) \in \mathcal{R}$, rule (r4) is not applicable, and $\forall R.(\ell_Q(x) \vee \ell_T(x)) \notin \ell(x)$, then we set $\ell(x) := \ell(x) \cup \{\forall R.(\ell_Q(x) \vee \ell_T(x))\}$;

(r6) if $(t_1 \vee t_2) \in \ell(x)$, for $t_i = (t_i^r, t_i^\forall, t_i^-)$, $i = 1, 2$, and there is no $j \in \{1, 2\}$ such that $t_j^\forall \subseteq \ell(x)$ and $t_j \in \mathfrak{m}(x)$, then take some $j \in \{1, 2\}$ and set $\ell(x) := \ell(x) \cup t_j^\forall$ and $\mathfrak{m}(x) := \mathfrak{m}(x) \cup \{t_j\}$.

4.3 RIs with Role Chains on the Left-Hand Side

The technique illustrated in the examples above works perfectly well for RIs with a single role in the left-hand side. To cope with more complex RIs, we follow Horrocks and Sattler (2004) and Horrocks et al. (2006) and encode every $R \in \mathit{role}(\mathcal{K})$ in a regular RBox $\mathcal{R}$ by means of a nondeterministic finite automaton (NFA) $\mathfrak{A}_R = (S_{\mathfrak{A}_R}, \mathit{role}(\mathcal{K}), s_R, \delta_{\mathfrak{A}_R}, a_R)$, where $S_{\mathfrak{A}_R}$ is a finite set of states, $\mathit{role}(\mathcal{K})$ is the input alphabet, $s_R \in S_{\mathfrak{A}_R}$ is the initial state of $\mathfrak{A}_R$, $\delta_{\mathfrak{A}_R}\colon S_{\mathfrak{A}_R} \times \mathit{role}(\mathcal{K}) \to 2^{S_{\mathfrak{A}_R}}$ is the transition function and $a_R \in S_{\mathfrak{A}_R}$ is the accepting state. If there are no REs in $\mathcal{R}$, then $\mathfrak{A}_R$ accepts precisely those role chains that belong to the language $L_{\mathcal{R}}(R)$; in other words $L(\mathfrak{A}_R) = L_{\mathcal{R}}(R)$.

In the tableau construction, whenever $\forall R.C \in \ell(x)$, we extend $\ell(x)$ with the quasi-concept $\forall \mathfrak{A}_R^s.C$, where s is the initial state of $\mathfrak{A}_R$. Next, if $\forall \mathfrak{A}_R^p.C \in \ell(x)$, y is a T-neighbour of x and $q \in \delta_{\mathfrak{A}_R}(p, T)$, then we extend $\ell(y)$ with $\forall \mathfrak{A}_R^q.C$. Finally, if $\forall \mathfrak{A}_R^a.C \in \ell(y)$, where a is an accepting state of $\mathfrak{A}_R$, we extend $\ell(y)$ with C. To define tableau rules more formally, we first confine attention to a single RI of the form $\boldsymbol{r} = (R \sqsubseteq QP)$.

We start by defining sets of quasi-concepts that are allowed for RBoxes containing $\boldsymbol{r}$. Denote by $\boldsymbol{qc}$ the set of all quasi-concepts of the form $\forall \mathfrak{A}_R^p.C$ such that $\forall R.C \in \mathit{con}(\mathcal{K})$

and p is a state of $\mathfrak{A}_R$. For a set $\Sigma \subseteq \boldsymbol{qc}$ and a basic role T, we now set:

$$\Sigma|_T^\forall = \{\forall \mathfrak{A}_R^q.C \mid \forall \mathfrak{A}_R^p.C \in \Sigma \text{ and } q \in \delta_{\mathfrak{A}_R}(p,T)\}, \tag{10}$$

$$\boldsymbol{qc}^*(\boldsymbol{r}) = \{\forall \mathfrak{A}_T^p.C \mid \forall \mathfrak{A}_T^p.C \in \boldsymbol{qc} \text{ and there exists } q \in \delta_{\mathfrak{A}_T}(p,Q)\} \ \cup \boldsymbol{qc}|_{Q^-}^\forall, \tag{11}$$

$$\boldsymbol{qc}(\boldsymbol{r}) = \big\{\forall \mathfrak{A}_R^p.\exists P^-.(t^r,t^\forall,t^-) \mid p \text{ a state of } \mathfrak{A}_R, t^r = Q,\ t^\forall \subseteq \boldsymbol{qc}|_Q^\forall,\ t^- \subseteq \boldsymbol{qc}|_{Q^-}^\forall\big\}. \tag{12}$$

It will be convenient to think of the labels $\ell(x)$ in tableaux as consisting of two disjoint parts $\ell(x) = \mathfrak{c}(x) \cup \mathfrak{a}(x)$, with $\mathfrak{c}(x)$ containing standard concepts and $\mathfrak{a}(x)$ quasi-concepts; that is: $\mathfrak{c}(x) \subseteq \mathit{con}(\mathcal{K})$ and

$$\mathfrak{a}(x) \ \subseteq\ \boldsymbol{qc} \cup \boldsymbol{qc}(\boldsymbol{r}) \cup \{\exists P^-.t \ \mid\ \forall \mathfrak{A}_R^p.\exists P^-.t \ \in\ \boldsymbol{qc}(\boldsymbol{r})\} \cup \{\neg\mathfrak{C} \ \mid\ \mathfrak{C} \ \in\ \boldsymbol{qc}^*(\boldsymbol{r})\}.$$

We now allow quasi-concepts of the form $\mathfrak{a}_Q(x) = (Q, \mathfrak{a}(x)|_Q^\forall, \mathfrak{a}(x) \cap \boldsymbol{qc}|_{Q^-}^\forall)$; we denote the first component of this triple by $\mathfrak{a}_Q^r(x)$, the second by $\mathfrak{a}_Q^\forall(x)$, and the third by $\mathfrak{a}_Q^-(x)$.

Using the new notation, we rewrite (r1)–(r3) as follows:

(r1) if $\boldsymbol{r} \in \mathcal{R}$ and there exists $\mathfrak{C} \in \boldsymbol{qc}^*(\boldsymbol{r})$ with $\{\mathfrak{C}, \neg\mathfrak{C}\} \cap \mathfrak{a}(x) = \emptyset$, then we set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{D}\}$, for some $\mathfrak{D} \in \{\mathfrak{C}, \neg\mathfrak{C}\}$;

(r2) if $\forall R.C \in \mathfrak{c}(x)$ and $\forall \mathfrak{A}_R^s.C \notin \mathfrak{a}(x)$, where s is the initial state of $\mathfrak{A}_R$, then we set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall \mathfrak{A}_R^s.C\}$;

(r3) if $\boldsymbol{r} \in \mathcal{R}$, rule (r1) is not applicable for $\boldsymbol{r}$ and $\forall \mathfrak{A}_R^s.\exists P^-.\mathfrak{a}_Q(x) \notin \mathfrak{a}(x)$, where s is the initial state of $\mathfrak{A}_R$, then we set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall \mathfrak{A}_R^s.\exists P^-.\mathfrak{a}_Q(x)\}$;

(r4) if $\forall \mathfrak{A}_R^p.\mathfrak{C} \in \mathfrak{a}(x)$, $q \in \delta_{\mathfrak{A}_R}(p,T)$, y is a T-neighbour of x and $\forall \mathfrak{A}_R^q.\mathfrak{C} \notin \mathfrak{a}(y)$, then we set $\mathfrak{a}(y) := \mathfrak{a}(y) \cup \{\forall \mathfrak{A}_R^q.\mathfrak{C}\}$;

(r5) if $\forall \mathfrak{A}_R^a.C \in \mathfrak{a}(x)$, a an accepting state, and $C \notin \mathfrak{c}(x)$, then we set $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{C\}$;

(r6) if $\forall \mathfrak{A}_R^a.\exists P^-.t \in \mathfrak{a}(x)$, where a is an accepting state, and $\exists P^-.t \notin \mathfrak{a}(x)$, then we set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\exists P^-.t\}$;

(r7) if $\exists P^-.t \in \mathfrak{a}(x)$, for $t = (t^r, t^\forall, t^-)$, and x has no P^--neighbour y with $t^\forall \subseteq \mathfrak{a}(y)$ and $t \in \mathfrak{m}(y)$, then we create a new P^--successor y of x and set $\mathfrak{a}(y) = t^\forall$ and $\mathfrak{m}(y) = \{t\}$.

The clash rule remains the same as before, with $\mathfrak{a}$ in place of ℓ. Note that the rule (r7) can be replaced with two rules such that one creates a new node y and sets $\mathfrak{a}(y) = \{t\}$, while the other rule sets $\mathfrak{a}(y) := \mathfrak{a}(y) \cup t^\forall$. In this case we do not need $\mathfrak{m}(y)$. We illustrate the new terminology and tableau rules by revisiting Example 11.

Example 11 (cont.) Consider again the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ with

$$\mathcal{A} = \{a : A\}, \quad \mathcal{R} = \{R \sqsubseteq QP\}, \quad \mathcal{T} = \{A \sqsubseteq \exists R.\top,\ A \sqsubseteq \forall Q.B,\ B \sqsubseteq \forall Q^-.C,\ C \sqsubseteq \forall Q.D\}.$$

In the tableau below, $\mathfrak{A}_Q$ and $\mathfrak{A}_R$ are NFAs with $L(\mathfrak{A}_Q) = \{Q\}$, $L(\mathfrak{A}_R) = \{R\}$, each having two states: initial s and accepting a.

$$\mathfrak{c}(x_0) = \{A, \exists R.\top, \forall Q.B\},\ \mathfrak{a}(x_0) = \{\forall \mathfrak{A}_Q^s.B\}, \qquad \text{(by } \sqsubseteq\text{, r2)}$$

$\mathfrak{a}(x_0) := \mathfrak{a}(x_0) \cup \{\forall\mathfrak{A}^s_Q.D, \forall\mathfrak{A}^a_{Q^-}.C\}$, $\mathfrak{c}(x_0) := \mathfrak{c}(x_0) \cup \{C\}$, (by r1, r5)

$\mathfrak{a}(x_0) := \mathfrak{a}(x_0) \cup \{\forall\mathfrak{A}^s_R.\exists P^-.\mathfrak{a}_Q(x_0)\}$, where
$\mathfrak{a}^\forall_Q(x_0) = \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B\}$, $\mathfrak{a}^-_Q(x_0) = \{\forall\mathfrak{A}^a_{Q^-}.C\}$, (by r3)

create x_1 with x_0Rx_1, $\mathfrak{c}(x_1) = \emptyset$, $\mathfrak{a}(x_1) = \{\forall\mathfrak{A}^s_Q.D, \forall\mathfrak{A}^s_Q.B, \forall\mathfrak{A}^a_{Q^-}.C\}$, (by $\exists R$, r1)

$\mathfrak{a}(x_1) := \mathfrak{a}(x_1) \cup \{\forall\mathfrak{A}^s_R.\exists P^-.\mathfrak{a}_Q(x_1)\}$, $\mathfrak{a}_Q(x_1) = (Q, \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B\}, \{\forall\mathfrak{A}^a_{Q^-}.C\})$, (by r3)

$\mathfrak{a}(x_1) := \mathfrak{a}(x_1) \cup \{\forall\mathfrak{A}^a_R.\exists P^-.\mathfrak{a}_Q(x_0), \exists P^-.\mathfrak{a}_Q(x_0)\}$, (by r4, r6)

create x_2 with $\mathfrak{m}(x_2) = \{\mathfrak{a}_Q(x_0)\}$, $x_1P^-x_2$, $\mathfrak{c}(x_2) = \emptyset$, $\mathfrak{a}(x_2) = \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B\}$, (by r7)

$\mathfrak{c}(x_2) := \{B, D, \forall Q^-.C\}$, $\mathfrak{a}(x_2) := \mathfrak{a}(x_2) \cup \{\forall\mathfrak{A}^s_{Q^-}.C\}$, (by r5, $\sqsubseteq$, r2)

$\mathfrak{a}(x_2) := \mathfrak{a}(x_2) \cup \{\forall\mathfrak{A}^s_Q.D, \forall\mathfrak{A}^s_Q.B, \neg\forall\mathfrak{A}^a_{Q^-}.C, \forall\mathfrak{A}^s_R.\exists P^-.(Q, \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B\}, \emptyset)\}$,
(by r1, r3)

As $\mathfrak{a}(x_2)|^\forall_{inv(Q)} = \{\forall\mathfrak{A}^a_{Q^-}.C\} \subseteq \mathfrak{a}^-_Q(x_0)$, the resulting tableau is complete and clash-free.

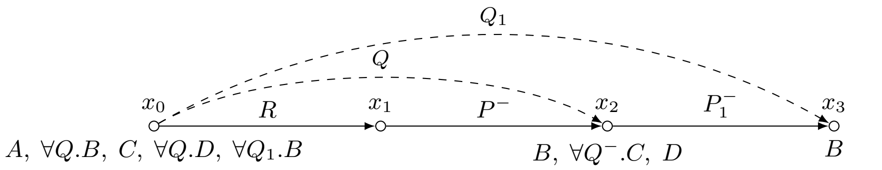
4.4 Interaction of RIs with Role Chains on the Right-Hand Side

Example 13 Consider the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ with

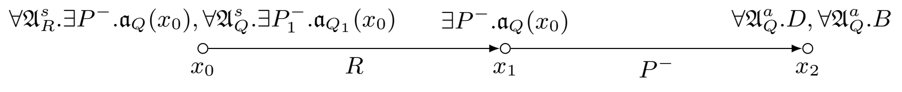
$$\mathcal{A} = \{a : A\}, \quad \mathcal{R} = \{R \sqsubseteq QP,\ Q \sqsubseteq Q_1P_1\},$$

$$\mathcal{T} = \{A \sqsubseteq \exists R.\top,\ A \sqsubseteq \forall Q.B,\ B \sqsubseteq \forall Q^-.C,\ C \sqsubseteq \forall Q.D,\ C \sqsubseteq \forall Q_1.B\}.$$

Here we have two RIs of the form (C), with Q occurring on the right-hand side of $R \sqsubseteq QP$ and in the left-hand side of $Q \sqsubseteq Q_1P_1$. We expect the tableau algorithm to construct a model of $\mathcal{K}$ as shown in the picture below:



However, if we apply the available rules, we can only produce the following tableau:



where $\mathfrak{a}_Q(x_0) = (Q, \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B\}, \{\forall\mathfrak{A}^a_{Q^-}.C\})$ and $\mathfrak{a}_{Q_1}(x_0) = (Q_1, \{\forall\mathfrak{A}^a_{Q_1}.B\}, \emptyset)$. As there is no explicit Q-arrow between x_0 and x_2, we cannot apply (r4) to obtain the quasi-concept $\forall\mathfrak{A}^a_Q.\exists P_1^-.\mathfrak{a}_{Q_1}(x_0)$, and so, by (r6), $\exists P_1^-.\mathfrak{a}_{Q_1}(x_0)$ in x_2, which would trigger the construction of a P_1^--arrow from x_2 to x_3. To overcome this problem, we will use the quasi-concept encoding the RI $Q \sqsubseteq Q_1P_1$ in the construction of the quasi-concept for $R \sqsubseteq QP$. More precisely, we add $\forall\mathfrak{A}^a_Q.\exists P_1^-.\mathfrak{a}_{Q_1}(x_0)$ to $\mathfrak{a}^\forall_Q(x_0)$, thus obtaining

$$\mathfrak{a}^\forall_Q(x_0) = \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B, \forall\mathfrak{A}^a_Q.\exists P_1^-.\mathfrak{a}_{Q_1}(x_0)\}.$$

As $\forall\mathfrak{A}^s_R.\exists P^-.\mathfrak{a}_Q(x_0)$ is in $\mathfrak{a}(x_0)$, we apply (r4) and obtain $\forall\mathfrak{A}^a_R.\exists P^-.\mathfrak{a}_Q(x_0)$, and so, by (r6), also $\exists P^-.\mathfrak{a}_Q(x_0)$ in $\mathfrak{a}(x_1)$. We then construct x_2 with $\mathfrak{a}(x_2)$ containing three quasi-concepts $\forall\mathfrak{A}^a_Q.D$, $\forall\mathfrak{A}^a_Q.B$ and $\forall\mathfrak{A}^a_Q.\exists P_1^-.\mathfrak{a}_{Q_1}(x_0)$, the last of which requires the existence of a P_1^--successor x_3.

In order to formalise the previous idea, we introduce a dependency relation $\lhd$. Given RIs $R_1 \sqsubseteq Q_1P_1$ and $R_2 \sqsubseteq Q_2P_2$, we write $(R_1 \sqsubseteq Q_1P_1) \lhd (R_2 \sqsubseteq Q_2P_2)$ if there are states p, q of $\mathfrak{A}_{R_1}$ such that either $q \in \delta_{\mathfrak{A}_{R_1}}(p, Q_2)$ or $q \in \delta_{\mathfrak{A}_{R_1}}(p, Q_2^-)$. In particular, we have $(Q \sqsubseteq Q_1P_1) \lhd (R \sqsubseteq QP)$. As $\mathcal{R}$ is regular, it is not hard to see that the relation $\lhd$ is acyclic. Indeed, it follows from the definition of $\lhd$, Lemma 1, Definition 3 and $\mathfrak{A}_{R_1}$ that $rn(Q_2) \prec_{\mathcal{R}} rn(Q_1)$ (since $rn(Q_2) \preceq_{\mathcal{R}} rn(R_1)$ and $rn(R_1) \prec_{\mathcal{R}} rn(Q_1)$).

Now, by induction on $\lhd$ we define sets $\boldsymbol{qc}(\boldsymbol{r})$ for RIs $\boldsymbol{r} = (R \sqsubseteq QP)$. For the $\lhd$-minimal $\boldsymbol{r}$, $\boldsymbol{qc}(\boldsymbol{r})$ is defined by (12). Then, assuming that $\boldsymbol{qc}(\boldsymbol{r}')$ is defined for every $\boldsymbol{r}' \lhd \boldsymbol{r}$ with $\boldsymbol{r} = (R \sqsubseteq QP)$, we set

$$\boldsymbol{qc}(\boldsymbol{r}) = \{\forall\mathfrak{A}^p_R.\exists P^-.t \mid p \text{ state in } \mathfrak{A}_R\},$$

where $t = (t^r, t^\forall, t^-)$, $t^r = Q$, $t^\forall \subseteq \boldsymbol{qc}|^\forall_Q \cup \bigcup_{\boldsymbol{r}'\lhd\boldsymbol{r}} \boldsymbol{qc}(\boldsymbol{r}')|^\forall_Q$, $t^- \subseteq \boldsymbol{qc}|^\forall_{Q^-} \cup \bigcup_{\boldsymbol{r}'\lhd\boldsymbol{r}} \boldsymbol{qc}(\boldsymbol{r}')|^\forall_{Q^-}$ and, for $\boldsymbol{r}' = (R' \sqsubseteq Q'P')$ and $\Sigma(\boldsymbol{r}') \subseteq \boldsymbol{qc}(\boldsymbol{r}')$,

$$\Sigma(\boldsymbol{r}')|^\forall_T = \{\forall\mathfrak{A}^q_{R'}.\mathfrak{C} \mid \forall\mathfrak{A}^p_{R'}.\mathfrak{C} \in \Sigma(\boldsymbol{r}') \text{ and } q \in \delta_{\mathfrak{A}_{R'}}(p, T)\}.$$

We also set

$$\begin{aligned}\boldsymbol{qc}^*(\boldsymbol{r}) = \{\forall\mathfrak{A}^p_T.\mathfrak{C} \mid \text{ there exists } q \in \delta_{\mathfrak{A}_T}(p, Q),\ \forall\mathfrak{A}^p_T.\mathfrak{C} \in \boldsymbol{qc} \cup \bigcup_{\boldsymbol{r}'\lhd\boldsymbol{r}} \boldsymbol{qc}(\boldsymbol{r}')\} \cup \\ \{\forall\mathfrak{A}^q_T.\mathfrak{C} \mid q \in \delta_{\mathfrak{A}_T}(p, Q^-),\ \forall\mathfrak{A}^p_T.\mathfrak{C} \in \boldsymbol{qc} \cup \bigcup_{\boldsymbol{r}'\lhd\boldsymbol{r}} \boldsymbol{qc}(\boldsymbol{r}')\}.\end{aligned}$$

Returning to our example, we see that $\boldsymbol{r}_1 \lhd \boldsymbol{r}$, for $\boldsymbol{r}_1 = (Q \sqsubseteq Q_1P_1)$, $\boldsymbol{r} = (R \sqsubseteq QP)$, and so $\boldsymbol{qc}(\boldsymbol{r}_1)$ remains as it was before, while $\boldsymbol{qc}(\boldsymbol{r})$ is becoming larger and, in particular, contains $\{\forall\mathfrak{A}^p_R.\exists P^-.(t_1^r, t_1^\forall, t_1^-) \mid p \in \{s, a\}\}$, where

$$t_1^\forall \subseteq \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B, \forall\mathfrak{A}^a_Q.\exists P_1^-.(Q_1, \{\forall\mathfrak{A}^a_{Q_1}.B\}, \emptyset), \forall\mathfrak{A}^a_Q.\exists P_1^-.(Q_1, \emptyset, \emptyset)\},\ t_1^- \subseteq \{\forall\mathfrak{A}^a_{Q^-}.C\}.$$

For example, $\boldsymbol{qc}(\boldsymbol{r})$ contains the quasi-concept

$$\forall\mathfrak{A}^s_R.\exists P^-.(Q, \{\forall\mathfrak{A}^a_Q.D, \forall\mathfrak{A}^a_Q.B, \forall\mathfrak{A}^a_Q.\exists P_1^-.(Q_1, \{\forall\mathfrak{A}^a_{Q_1}.B\}, \emptyset)\}, \{\forall\mathfrak{A}^a_{Q^-}.C\}).$$

The construction of a tableau for $\mathcal{K}$ in Example 13, using the newly defined sets $\boldsymbol{qc}(\boldsymbol{r})$, is routine and left to the reader.

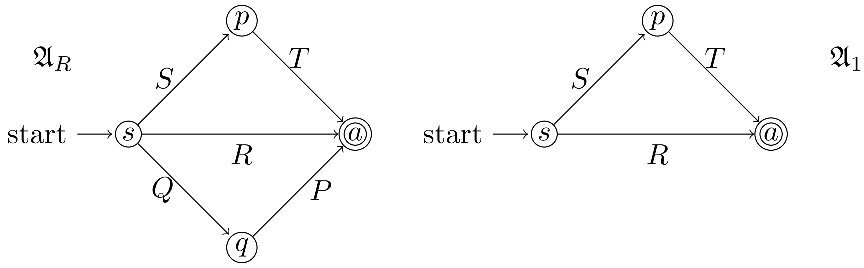
The dependency relation $\lhd$ between RIs in RBoxes will become more complex in the presence of unions of roles.

4.5 Role Equalities

Example 14 Consider the RBox $\mathcal{R}$ with two RAs: $ST \sqsubseteq R$ of type (B) and $R = QP$ of type (E). Clearly, $R = QP$ can be replaced by the RIs $R \sqsubseteq QP$ and $QP \sqsubseteq R$, but the resulting RBox $\{ST \sqsubseteq R,\ R \sqsubseteq QP,\ QP \sqsubseteq R\}$ will not be regular. Let us observe now that both RBoxes

$$\mathcal{R}' = \{ST \sqsubseteq R,\ QP \sqsubseteq R\} \quad \text{and} \quad \mathcal{R}'' = \{ST \sqsubseteq R,\ R \sqsubseteq QP\}$$

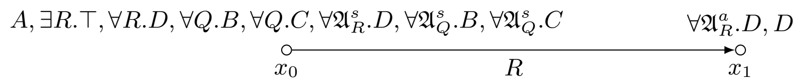
are regular. Denote by $\mathfrak{A}_R$ the NFA for R determined by $\mathcal{R}'$, and by $\mathfrak{A}_1$ the NFA for R given by $\mathcal{R}''$ (see the picture below).



Let us see now whether we can use any of these automata in the rules (r2) and (r3) on page 823 for the role R. Consider the KB $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$, where $\mathcal{R}$ is as above and

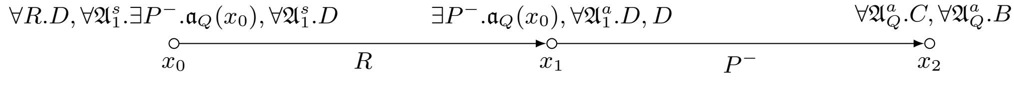
$$\mathcal{A} = \{a : A\}, \quad \mathcal{T} = \{A \sqsubseteq \exists R.\top,\ A \sqsubseteq \forall Q.B,\ A \sqsubseteq \forall Q.C,\ A \sqsubseteq \forall R.D\}.$$

First we try $\mathfrak{A}_R$. By applying the tableau rules we obtain the following:



Now we have to apply (r3) to the RI $R \sqsubseteq QP$ and add the quasi-concept $\forall \mathfrak{A}^s_R.\exists P^-.\mathfrak{a}_Q(x_0)$ to $\mathfrak{a}(x_0)$, where $\mathfrak{a}_Q(x_0) = (Q, \mathfrak{a}(x_0)|^{\forall}_{Q}, \mathfrak{a}(x_0) \cap \boldsymbol{qc}|^{\forall}_{Q^-})$. Because of the Q-transition in $\mathfrak{A}_R$, we must then have $\forall \mathfrak{A}^q_R.\exists P^-.\mathfrak{a}_Q(x_0) \in \mathfrak{a}(x_0)|^{\forall}_{Q}$, which is impossible because $\mathfrak{a}(x_0)|^{\forall}_{Q}$ cannot be an element of itself.

Alternatively, we can use $\mathfrak{A}_1$ for R. This gives us



with $\mathfrak{a}_Q(x_0) = (Q, \{\forall \mathfrak{A}^a_Q.C, \forall \mathfrak{A}^a_Q.B\}, \emptyset)$, which defines a model of $\mathcal{K}$ when we add the missing Q-arrow from x_0 to x_2.

Now, we replace the CI $A \sqsubseteq \exists R.\top$ in $\mathcal{K}$ with $A \sqsubseteq \exists Q.\exists P.\top$ and use $\mathfrak{A}_1$ for R. In this case, we obtain the following tableau:

To produce a satisfying interpretation $\mathcal{I}$, we have to add an R-arrow from x_0 to x_2. However, this cannot be done 'for free' (as in Example 10) because $x_2 \notin D^{\mathcal{I}}$. An alternative would be to use $\mathfrak{A}_R$ and $\mathcal{R}'$ instead of $\mathcal{R}$ (because we do not have to apply (r3) to $R \sqsubseteq QP$). We then obtain the following tableau:

$$\begin{tikzcd} \underset{x_0}{\overset{\forall R.D, \forall \mathfrak{A}^s_R.D}{\circ}} \arrow[r, "Q"'] & \underset{x_1}{\overset{\forall\mathfrak{A}^q_R.D}{\circ}} \arrow[r, "P"'] & \underset{x_2}{\overset{\forall\mathfrak{A}^a_R.D, D}{\circ}} \end{tikzcd}$$

The addition of an R-arrow from x_0 to x_2 gives an interpretation $\mathcal{I}$ such that $\mathcal{I} \models QP \sqsubseteq R$ and $x_2 \in D^{\mathcal{I}}$.

To sum up: the rule (r2) requires the NFA $\mathfrak{A}_R$, while (r3) requires $\mathfrak{A}_1$. So rule (r2) on page 823 remains the same and we rewrite rule (r1) and (r3) to include role equalities as follows:

(r1) if $\boldsymbol{r} \in \mathcal{R}$, for $\boldsymbol{r} = (R \sqsubseteq QP)$ or $\boldsymbol{r} = (R = QP)$, and there exists $\mathfrak{C} \in \boldsymbol{qc}^*(\boldsymbol{r})$ with $\{\mathfrak{C}, \neg\mathfrak{C}\} \cap \mathfrak{a}(x) = \emptyset$, then we set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{D}\}$, for some $\mathfrak{D} \in \{\mathfrak{C}, \neg\mathfrak{C}\}$;

(r3) if $\boldsymbol{r} \in \mathcal{R}$, for $\boldsymbol{r} = (R \sqsubseteq QP)$ or $\boldsymbol{r} = (R = QP)$, rule (r1) is not applicable for $\boldsymbol{r}$ and $\forall\mathfrak{A}_1^s.\exists P^-.\mathfrak{a}_Q(x) \notin \mathfrak{a}(x)$, where s is the initial state of $\mathfrak{A}_1$, then we set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall\mathfrak{A}_1^s.\exists P^-.\mathfrak{a}_Q(x)\}$.

Note that in the case $\boldsymbol{r} = (R \sqsubseteq QP)$ NFA $\mathfrak{A}_1$ is same as $\mathfrak{A}_R$ and in the case $\boldsymbol{r} = (R = QP)$ NFA $\mathfrak{A}_1$ is different from $\mathfrak{A}_R$ as described above.

5. Main Result and Discussion

The examples of the previous section provide the basic ingredients that can be added to $\mathcal{SROIQ}$ tableaux of Horrocks et al. (2006) and Horrocks and Sattler (2007) in order to obtain sound and complete tableaux for $\mathcal{SR}^+\mathcal{OIQ}$. We present all the technical details and definitions in Appendix A. A corresponding sound, complete and terminating tableau algorithm is given in Appendix B. Thus, we obtain the following:

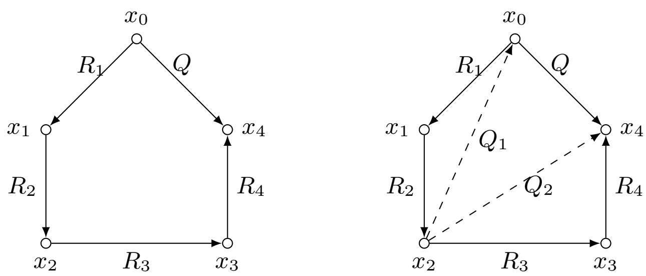
Theorem 15 *Concept satisfiability with respect to $\mathcal{SR}^+\mathcal{OIQ}$ KBs is decidable.*

It is to be noted that the decision algorithm in Appendix B is a (quite sophisticated) extension of the standard tableau procedure for $\mathcal{SROIQ}$; if the input RBox does not contain RAs of the form (C)–(F) then our tableau algorithm behaves exactly as the $\mathcal{SROIQ}$ procedure. To simplify presentation and avoid a number of technical details, we decided not to optimise our tableau algorithm in this paper. In fact, there is plenty of room for optimisations; for example, one can work on a more careful choice of quasi-concepts as well as utilise the approach of Motik, Shearer, and Horrocks (2009).

The exact complexity of concept satisfiability with respect to $\mathcal{SR}^+\mathcal{OIQ}$ KBs is still unknown. If the RBox contains one RA $\boldsymbol{r}_1$ of the form (C)–(F), our algorithm will have to construct the set $\boldsymbol{qc}(\boldsymbol{r}_1)$ of quasi-concepts, which contains subsets of the previously constructed sets of quasi-concepts $\boldsymbol{qc}(\boldsymbol{r}_0)$, and so may suffer an exponential blow-up. Furthermore, the algorithm may suffer one more exponential blow-up every time we add an extra RA of the form (C)–(F), and thereby extend the $\lhd$-chains of RAs, because again the set of quasi-concepts may become exponentially larger. To investigate the complexity of full $\mathcal{SR}^+\mathcal{OIQ}$, it may be useful to consider first its various sub-languages. For example, we conjecture that $\mathcal{ALCI}$-concept satisfiability with respect to regular RBoxes that only contain axioms of type (C) and the roles $rn(R_i)$, $i = 1, \dots, m$, do not appear in left-hand side of RIs, is PSpace-complete. $\mathcal{SI}$-concept satisfiability with respect to RBoxes which contain only one axiom of the form $R \sqsubseteq QP$, where $rn(R), rn(Q), rn(P)$ are different role names that are not transitive, is also PSpace-complete.

The step from $\mathcal{SROIQ}$ to $\mathcal{SR}^+\mathcal{OIQ}$ is, to some extent, similar to the step from $\mathcal{SHOIQ}$ to $\mathcal{SROIQ}$: as $\mathcal{SROIQ}$ extends $\mathcal{SHOIQ}$ with role inclusion axioms containing role chains in the *left-hand* side, $\mathcal{SR}^+\mathcal{OIQ}$ extends $\mathcal{SROIQ}$ with role inclusion axioms containing role chains or unions in the *right-hand* side. Attempts to extend various DLs with such role inclusions have been made since 1985 (Brachman & Schmolze, 1985; Baader, 2003; Wessel, 2001, 2002); however, all of them resulted in undecidable formalisms. Similar problems were investigated in modal logic, where it was shown that regular grammar logics are decidable (Demri, 2001). Our regularity condition for RAs axioms generalises the restrictions of Horrocks et al. (2006) and Kazakov (2010). (However, a closer inspection of how our results are related to grammar modal logics is needed.) Simančík (2012) showed that complex RIs in $\mathcal{SROIQ}$ can be encoded using $\mathcal{SHOIQ}$ axioms. It would be of interest to find out whether a similar reduction is possible in the case of $\mathcal{SR}^+\mathcal{OIQ}$.

One of the aims of introducing complex role inclusion axioms in DLs is to model complex structured objects. Suppose, for example, that we have to represent the cycle shown on the left-hand side of the picture below:



In $\mathcal{SROIQ}$, we can only use the RI axiom $R_1R_2R_3R_4 \sqsubseteq Q$, which produces the required cycle only if there is a chain of the form $R_1R_2R_3R_4$. Using description graphs from the work of Motik et al. (2009), we can express the existence of the cycle above as a whole. In $\mathcal{SR}^+\mathcal{OIQ}$, we can model this situation by the following regular RBox, where Q_1 and Q_2 are fresh role names:

$$R_1 \sqsubseteq QR_4^-R_3^-R_2^-, \qquad R_2^- \sqsubseteq Q_1R_1, \qquad Q_1^- \sqsubseteq QR_4^-R_3^-,$$
$$R_3 \sqsubseteq Q_2R_4^-, \qquad Q_2^- \sqsubseteq Q^-R_1R_2, \qquad R_4 \sqsubseteq Q^-R_1R_2R_3$$

(see the picture above). The RBox produces the required cycle if there is at least one R_i, for $i = 1, 2, 3, 4$, in a model. In this connection, it would be of interest to consider the extension of $\mathcal{SHOIQ}$ with RI axioms of the form (A) and (C).

Appendix A. $\mathcal{SR}^+\mathcal{OIQ}$ Tableaux

As observed by Horrocks et al. (2006, Thm. 9), without loss of generality we can define tableaux for $\mathcal{SR}^+\mathcal{OIQ}$ KBs with empty TBoxes and ABoxes. Let $\mathcal{R}$ be a regular RBox and C_0 a $\mathcal{SR}^+\mathcal{OIQ}$ concept. We assume that $\mathcal{R}_{C,D,E,F} = \{\boldsymbol{r}_i \mid i = 1,\dots,l\}$, where, for some k_1, k, l_1 such that $1 \leq k_1 \leq k \leq l_1 \leq l$,

$$\begin{aligned}
\boldsymbol{r}_i &= (R_i \sqsubseteq Q_i P_{i1} \dots P_{im_i}), && \text{for } i = 1,\dots,k_1,\\
\boldsymbol{r}_i &= (R_i = Q_i P_{i1} \dots P_{im_i}), && \text{for } i = k_1+1,\dots,k,\\
\boldsymbol{r}_i &= (R_i = T_{i1} \sqcup \dots \sqcup T_{im_i}), && \text{for } i = k+1,\dots,l_1,\\
\boldsymbol{r}_i &= (R_i \sqsubseteq T_{i1} \sqcup \dots \sqcup T_{im_i}), && \text{for } i = l_1+1,\dots,l.
\end{aligned}$$

For every $R \in \mathit{role}(C_0,\mathcal{R})$, we construct, as a preprocessing step, an NFA $\mathfrak{A}_R$ and special NFAs $\mathfrak{A}_i$, for $i = 1,\dots,l$, as described below. Recall that $L(\mathfrak{A})$ denotes the language recognised by $\mathfrak{A}$. If p is a state in $\mathfrak{A}$, then $\mathfrak{A}^p$ is the NFA obtained from $\mathfrak{A}$ by making p the (only) initial state of $\mathfrak{A}$.

Define an RBox

$$\mathcal{R}' = \mathcal{R}_{A,B} \cup \{Q_i P_{i1} \dots P_{im_i} \sqsubseteq R_i \mid i = k_1+1,\dots,k\} \cup \{T_{ij} \sqsubseteq R_i \mid i = k+1,\dots,l_1,\ j = 1,\dots,m_i\},$$

which only contains axioms of types (A) and (B). Since $\mathcal{R}$ is regular and in view of conditions **(c1)**, **(c4)** and **(c6)** in Definition 3, the RBox $\mathcal{R}'$ is stratified. By Theorem 2, we use $\mathcal{R}'$ to construct, for any $R \in \mathit{role}(C_0,\mathcal{R})$, an NFA $\mathfrak{A}_R = (S_{\mathfrak{A}_R}, \mathit{role}(C_0,\mathcal{R}), s_R, \delta_{\mathfrak{A}_R}, a_R)$ such that $L(\mathfrak{A}_R) = L_{\mathcal{R}'}(R)$.

We also define RBoxes

$$\begin{aligned}
\mathcal{R}^i &= \mathcal{R}' \setminus \{Q_i P_{i1} \dots P_{im_i} \sqsubseteq R_i\}, && i = k_1+1,\dots,k,\\
\mathcal{R}^i &= \mathcal{R}' \setminus \{T_{ij} \sqsubseteq R_i \mid j = 1,\dots,m_i\}, && i = k+1,\dots,l_1,
\end{aligned}$$

and construct NFAs $\mathfrak{A}_i$ such that $L(\mathfrak{A}_i) = L_{\mathcal{R}^i}(R_i)$, $i = k_1+1,\dots,l_1$. For $i = 1,\dots,k_1$ and $i = l_1+1,\dots,l$, we simply set $\mathfrak{A}_i = \mathfrak{A}_{R_i}$.

Now, we are going to define formally the set $\boldsymbol{qc}(C_0,\mathcal{R})$. The elements of $\boldsymbol{qc}(C_0,\mathcal{R})$ are called *quasi-concepts* (for C_0 w.r.t. $\mathcal{R}$); we use them to define labels for tableau nodes. In the definition of $\boldsymbol{qc}(C_0,\mathcal{R})$, we require a dependency relation $\lhd$ on $\mathcal{R}_{C,D,E,F}$.

For each role name $Q \in \{rn(Q_i),\ rn(T_{i1}),\dots,rn(T_{im_i}) \mid 1 \leq i \leq l\}$, let $\mathit{AutIn}(Q)$ be the set of those $i \in \{1,\dots,l\}$ for which there are states p and q of $\mathfrak{A}_i$ such that $q \in \delta_{\mathfrak{A}_i}(p,Q)$ or $q \in \delta_{\mathfrak{A}_i}(p,Q^-)$. We define $\lhd$ on $\mathcal{R}_{C,D,E,F}$ by taking $\boldsymbol{r}_i \lhd \boldsymbol{r}_j$ if

- $1 \leq j \leq k$ and $i \in \mathit{AutIn}(rn(Q_j))$, or
- $k < j \leq l$ and there is $h \in \{1,\dots,m_j\}$ such that $i \in \mathit{AutIn}(rn(T_{jh}))$.

The following lemma shows that the transitive closure of $\lhd$ is acyclic:

Lemma 16 (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.*

Proof. Observe first that if $i \in \mathit{AutIn}(Q)$ then there is a Q- or Q^--transition of $\mathfrak{A}_i$, and so we have $rn(Q) \preceq^1_{\mathcal{R}} rn(R_i)$. If $i \le k$ then $rn(R_i) \prec^1_{\mathcal{R}} rn(Q_i)$, and so $rn(Q) \prec^1_{\mathcal{R}} rn(Q_i)$. If $i > k$ then $rn(R_i) \prec^1_{\mathcal{R}} rn(T_{ih})$, and so $rn(Q) \prec^1_{\mathcal{R}} rn(T_{ih})$, for all $h \in \{1, \dots, m_i\}$.

(i) Let $\boldsymbol{r}_i \lhd \boldsymbol{r}_j$. Four cases are possible.

Case 1: $i, j \le k$. Then $i \in \mathit{AutIn}(rn(Q_j))$, and so $rn(Q_j) \prec^1_{\mathcal{R}} rn(Q_i)$. Similarly, if we had $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$, then $rn(Q_i) \prec^1_{\mathcal{R}} rn(Q_j)$, which is impossible.

Case 2: $j \le k$ and $i > k$. Then $i \in \mathit{AutIn}(rn(Q_j))$, and so $rn(Q_j) \prec^1_{\mathcal{R}} rn(T_{ih})$, for all $h \in \{1, \dots, m_i\}$. If $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$ then there is T_{ih_0}, $1 \le h_0 \le m_i$, such that $j \in \mathit{AutIn}(rn(T_{ih_0}))$. Hence, $rn(T_{ih_0}) \prec^1_{\mathcal{R}} rn(Q_j)$, which is a contradiction.

Case 3: $i \le k$ and $j > k$. This is a mirror image of case 2.

Case 4: $i, j > k$. Then there is T_{jh_0}, $1 \le h_0 \le m_j$, such that $i \in \mathit{AutIn}(rn(T_{jh_0}))$, and so $rn(T_{jh_0}) \prec^1_{\mathcal{R}} rn(T_{ie})$, for all $e \in \{1, \dots, m_i\}$. Similarly, if we had $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$, then there is T_{ie_0}, $1 \le e_0 \le m_i$, such that $rn(T_{ie_0}) \prec^1_{\mathcal{R}} rn(T_{jh})$, for all $h \in \{1, \dots, m_j\}$, which is impossible.

The proof of (ii) is similar and left to the reader. ❑

We will require the following notation. Let

$$\boldsymbol{qc} = \{\forall \mathfrak{A}^p_R.C \mid \forall R.C \in con(C_0) \text{ and } p \text{ a state of } \mathfrak{A}_R\}.$$

For a set $\Sigma \subseteq \boldsymbol{qc}$ and a basic role P, we set

$$\Sigma|^{\forall}_P = \{\forall \mathfrak{A}^q_R.C \mid \forall \mathfrak{A}^p_R.C \in \Sigma \text{ and } q \in \delta_{\mathfrak{A}_R}(p, P)\}.$$

Sometimes it will be convenient for us to write $\boldsymbol{qc}(\boldsymbol{r}_0)$ in place of $\boldsymbol{qc}$ and assume that $\boldsymbol{r}_0 \lhd \boldsymbol{r}_i$, for all i, $1 \le i \le l$. Now, assuming that $\boldsymbol{qc}(\boldsymbol{r}_j)$ is defined for every $\boldsymbol{r}_j \lhd \boldsymbol{r}_i$ where $0 \le j \le l$ and $1 \le i \le l$, we define $\boldsymbol{qc}(\boldsymbol{r}_i)$ to be the set of all $\forall \mathfrak{A}^q_i.\mathfrak{C}$ such that

- q is a state of $\mathfrak{A}_i$;
- for $i \le k$, $\mathfrak{C} = \exists P^-_{im_i} \cdot \dots \exists P^-_{i1}.(t^r_0, t^{\forall}_0, t^-_0)$ and $t^r_0 = Q_i$;
- for $i > k$, $\mathfrak{C} = \bigvee_{h=1}^{m_i} (t^r_h, t^{\forall}_h, t^-_h)$ and $t^r_h = T_{ih}$;
- $t^{\forall}_h \subseteq \bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)|^{\forall}_{t^r_h}$;
- $t^-_h \subseteq \bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)|^{\forall}_{inv(t^r_h)}$.

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

$$\Sigma(\boldsymbol{r}_i)|^{\forall}_P = \{\forall \mathfrak{A}^q_i.\mathfrak{C} \mid \forall \mathfrak{A}^p_i.\mathfrak{C} \in \Sigma(\boldsymbol{r}_i) \text{ and } q \in \delta_{\mathfrak{A}_i}(p, P)\}.$$

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,

$$\Sigma|^{\forall}_P = \bigcup_{j=0}^{l} (\Sigma \cap \boldsymbol{qc}(\boldsymbol{r}_j))|^{\forall}_P \qquad \text{and} \qquad \Sigma|^{\forall}_{\varepsilon} = \{\forall \mathfrak{A}^q.\mathfrak{C} \mid \forall \mathfrak{A}^p.\mathfrak{C} \in \Sigma \text{ and } q \in \delta_{\mathfrak{A}}(p, \varepsilon)\},$$

where ε is the empty role chain. For $\Sigma \subseteq \boldsymbol{qc}(C_0, \mathcal{R})$ and $1 \leq i \leq l$, let

$$\Xi(\boldsymbol{r}_i, \Sigma) = \begin{cases} \exists P^-_{im_i} \dots \exists P^-_{i1}.(t^r_0, t^\forall_0, t^-_0),\ t^r_0 = Q_i, & \text{if } i \leq k, \\ \bigvee_{h=1}^{m_i}(t^r_h, t^\forall_h, t^-_h),\ t^r_h = T_{ih}, & \text{if } i > k, \end{cases} \tag{13}$$

where $t^\forall_h = \Sigma|^\forall_{t^r_h}$, $t^-_h = \Sigma \cap \boldsymbol{qc}(C_0, \mathcal{R})|^\forall_{inv(t^r_h)}$, $0 \leq h \leq m_i$. Clearly, $\forall\mathfrak{A}^s_i.(\Xi(\boldsymbol{r}_i, \Sigma)) \in \boldsymbol{qc}(\boldsymbol{r}_i)$. Intuitively, if Σ is the label of a node u, that is, $\Sigma = \mathfrak{a}(u)$, then $\Xi(\boldsymbol{r}_i, \Sigma)$ is the quasi-concept encoding the RA $\boldsymbol{r}_i$ in the node u.

Example 17 Let $\mathcal{R} = \{\boldsymbol{r}_1, \boldsymbol{r}_2\}$, where $\boldsymbol{r}_1 = (R_1 \sqsubseteq Q_1 P_1 P_2)$ and $\boldsymbol{r}_2 = (R_2 \sqsubseteq T_1 \sqcup T_2)$. The NFAs for the roles in $\mathcal{R}$ have two states: initial s and accepting a. Suppose

$$\Sigma = \{\forall\mathfrak{A}^s_{Q_1}.C_1,\ \forall\mathfrak{A}^a_{Q_1^-}.C_2,\ \forall\mathfrak{A}^s_{T_1}.C_3,\ \forall\mathfrak{A}^s_{T_2}.C_4, \forall\mathfrak{A}^s_{T_2}.C_5,\ \forall\mathfrak{A}^a_{T_1^-}.C_6\}.$$

Then

$$\Xi(\boldsymbol{r}_1, \Sigma) = \exists P_2^-.\exists P_1^-.(Q_1, \{\forall\mathfrak{A}^a_{Q_1}.C_1\}, \{\forall\mathfrak{A}^a_{Q_1^-}.C_2\}),$$
$$\Xi(\boldsymbol{r}_2, \Sigma) = (T_1, \{\forall\mathfrak{A}^a_{T_1}.C_3\}, \{\forall\mathfrak{A}^a_{T_1^-}.C_6\}) \vee (T_2, \{\forall\mathfrak{A}^a_{T_2}.C_4, \forall\mathfrak{A}^a_{T_2}.C_5\}, \emptyset).$$

Remark 18 If P is a symmetric role (i.e., $(P^- \sqsubseteq P) \in \mathcal{R}$), then each occurrence of P and $inv(P)$ is treated as $rn(P)$. For example, $\{\forall P.D, \forall P^-.C\}|^\forall_{P^-} = \{D, C\}$.

We are now in a position to define $\mathcal{SR}^+\mathcal{OIQ}$ tableaux. Note that the most essential difference compared with the tableaux for $\mathcal{SROIQ}$ (Horrocks et al., 2006) are the rules **(p19)**, **(p21)** and **(p22)**.

A *tableau* for C_0 w.r.t. $\mathcal{R}$ is a structure of the form $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}, \mathfrak{a}, \mathcal{E})$, where $\boldsymbol{S}$ is non-empty set, $\mathfrak{c}\colon \boldsymbol{S} \to 2^{con(C_0)}$, $\mathfrak{a}\colon \boldsymbol{S} \to 2^{\boldsymbol{qc}(C_0,\mathcal{R})}$, $\mathcal{E}\colon role(C_0, \mathcal{R}) \to 2^{\boldsymbol{S}\times\boldsymbol{S}}$ such that the following conditions hold:

(p1) $C_0 \in \mathfrak{c}(u_0)$ for some $u_0 \in \boldsymbol{S}$,

(p2) if $C \in \mathfrak{c}(u)$ then $\neg C \notin \mathfrak{c}(u)$, where C is either a concept name or $\exists R.Self$,

(p3) $\top \in \mathfrak{c}(u)$ and $\bot \notin \mathfrak{c}(u)$ for any u,

(p4) if $\exists R.Self \in \mathfrak{c}(u)$ then $(u, u) \in \mathcal{E}(R)$,

(p5) if $\neg\exists R.Self \in \mathfrak{c}(u)$ then $(u, u) \notin \mathcal{E}(R)$,

(p6) if $(C_1 \sqcap C_2) \in \mathfrak{c}(u)$ then $C_1 \in \mathfrak{c}(u)$ and $C_2 \in \mathfrak{c}(u)$,

(p7) if $(C_1 \sqcup C_2) \in \mathfrak{c}(u)$ then $C_1 \in \mathfrak{c}(u)$ or $C_2 \in \mathfrak{c}(u)$,

(p8) if $\exists R.C \in \mathfrak{c}(u)$ then there is some $v \in \boldsymbol{S}$ with $(u, v) \in \mathcal{E}(R)$ and $C \in \mathfrak{c}(v)$,

(p9) $(u, v) \in \mathcal{E}(R)$ iff $(v, u) \in \mathcal{E}(inv(R))$,

(p10) if $(\leq nS.C) \in \mathfrak{c}(u)$ then $\sharp\{v \in \boldsymbol{S} \mid (u, v) \in \mathcal{E}(S) \text{ and } C \in \mathfrak{c}(v)\} \leq n$,

(p11) if $(\geq nS.C) \in \mathfrak{c}(u)$ then $\sharp\{v \in \boldsymbol{S} \mid (u, v) \in \mathcal{E}(S)$ and $C \in \mathfrak{c}(v)\} \geq n$,

(p12) if $(\leq nS.C) \in \mathfrak{c}(u)$ and $(u, v) \in \mathcal{E}(S)$, then $C \in \mathfrak{c}(v)$ or $\neg C \in \mathfrak{c}(v)$,

(p13) if $o \in \mathfrak{c}(u) \cap \mathfrak{c}(v)$, for some $o \in nom(C_0)$, then $v = u$,

(p14) for each $o \in nom(C_0)$, there is some $v_o \in \boldsymbol{S}$ with $o \in \mathfrak{c}(v_o)$,

(p15) if $Dis(R, S) \in \mathcal{R}$ then $\mathcal{E}(R) \cap \mathcal{E}(S) = \emptyset$,

(p16) if $(u, v) \in \mathcal{E}(R)$ and $R \sqsubseteq^* S$, then $(u, v) \in \mathcal{E}(S)$,[2]

(p17) if $\forall R.C \in \mathfrak{c}(u)$ then $\forall \mathfrak{A}_R^s.C \in \mathfrak{a}(u)$, where s is the initial state of $\mathfrak{A}_R$,

(p18) if $\forall \mathfrak{A}_R^a.C \in \mathfrak{a}(u)$, where a is an accepting state, then $C \in \mathfrak{c}(u)$,

(p19) $\forall \mathfrak{A}_i^s.\mathfrak{C} \in \mathfrak{a}(u)$, where s is the initial state of $\mathfrak{A}_i$ and $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \mathfrak{a}(u))$, for all $u \in \boldsymbol{S}$ and $1 \leq i \leq l$,

(p20) if $(u, v) \in \mathcal{E}(R)$ then $\mathfrak{a}(u)|_R^\forall \subseteq \mathfrak{a}(v)$,

(p21) if $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(u)$, where $i \leq k$, $\mathfrak{C} = \exists inv(P_{im_i}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-)$ and a is an accepting state, then there are $v_0, v_1, \ldots, v_{m_i} = u$ such that $(v_j, v_{j-1}) \in \mathcal{E}(inv(P_{ij}))$, for $1 \leq j \leq m_i$, $t^\forall \subseteq \mathfrak{a}(v_0)$ and $\mathfrak{a}(v_0)|_{inv(t^r)}^\forall \subseteq t^-$,

(p22) if $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(u)$, where a is an accepting state, $i > k$ and $\mathfrak{C} = \bigvee_{h=1}^{m_i}(t_h^r, t_h^\forall, t_h^-)$, then there is $j \in \{1, \ldots, m_i\}$ such that $t_j^\forall \subseteq \mathfrak{a}(u)$ and $\mathfrak{a}(u)|_{inv(t_j^r)}^\forall \subseteq t_j^-$,

(p23) $\mathfrak{a}(u)|_\varepsilon^\forall \subseteq \mathfrak{a}(u)$.

Let $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}, \mathfrak{a}, \mathcal{E})$ be a tableau, R a basic role and $u, v \in \boldsymbol{S}$. If $\mathfrak{a}(u)|_R^\forall \subseteq \mathfrak{a}(v)$ and $\mathfrak{a}(v)|_{inv(R)}^\forall \subseteq \mathfrak{a}(u)$, then we write $ar(R, u, v)$. If there is an R-arrow from u to v then $ar(R, u, v)$ holds; see Proposition 20 (i). On the other hand, the meaning of $ar(R, u, v)$ is that we can always insert an R-arrow (for a non-simple role R) from u to v without violating any of the tableau conditions.

Lemma 19 *A concept C_0 is satisfiable w.r.t. a $\mathcal{SR}^+\mathcal{OIQ}$ RBox $\mathcal{R}$ if and only if there exists a tableau for C_0 w.r.t. $\mathcal{R}$.*

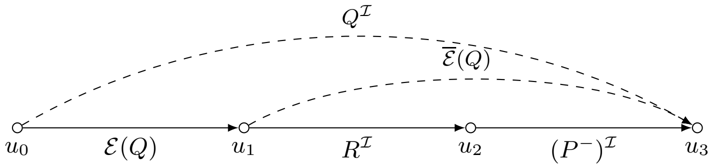
Proof. ($\Leftarrow$) Let $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}, \mathfrak{a}, \mathcal{E})$ be a tableau for C_0 w.r.t. $\mathcal{R}$. Define an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ by taking $\Delta^\mathcal{I} = \boldsymbol{S}$, $C^\mathcal{I} = \{u \mid C \in \mathfrak{c}(u)\}$, for a concept name $C \in con(C_0)$. For a role name R, we define $\overline{\mathcal{E}}(R)$ (by induction on $\prec_\mathcal{R}^1$) and $R^\mathcal{I}$ in the following way. For a role name R, we define $\overline{\mathcal{E}}(R)$ and $R^\mathcal{I}$ by induction on $\prec_\mathcal{R}^1$ in the following way. For $\prec_\mathcal{R}^1$-minimal R, we set $\overline{\mathcal{E}}(R) = \mathcal{E}(R)$. We extend $\overline{\mathcal{E}}(\cdot)$ with $\overline{\mathcal{E}}(inv(R)) = \{(u, v) | (v, u) \in \overline{\mathcal{E}}(R)\}$

2. Here $\sqsubseteq^*$ is the transitive closure of $\sqsubseteq$.

and $\overline{\mathcal{E}}(S_1 \dots S_n) = \overline{\mathcal{E}}(S_1) \circ \dots \circ \overline{\mathcal{E}}(S_n)$. Suppose now that $\overline{\mathcal{E}}(S)$ is defined for all $S \prec^1_{\mathcal{R}} R$. Then we set, where $w_i = P_{i1} \dots P_{im_i}$ and $\mathcal{E}(w_i) = \mathcal{E}(P_{i1}) \circ \dots \circ \mathcal{E}(P_{im_i})$,

$$
\begin{aligned}
\overline{\mathcal{E}}(R) \;=\; & \mathcal{E}(R) \;\cup \\
& \bigcup_{\{i \mid R = Q_i\}} \{(u,v) \mid ar(R,u,v) \;\&\; \exists \varrho \in L(\mathfrak{A}_i)\, \exists z\, ((u,z) \in \overline{\mathcal{E}}(\varrho) \wedge (v,z) \in \mathcal{E}(w_i))\} \;\cup \\
& \bigcup_{\{i \mid R = inv(Q_i)\}} \{(u,v) \mid ar(R,u,v) \;\&\; \exists \varrho \in L(\mathfrak{A}_i)\, \exists z\, ((v,z) \in \overline{\mathcal{E}}(\varrho) \wedge (u,z) \in \mathcal{E}(w_i))\} \;\cup \\
& \bigcup_{\{i \mid \exists j\, R = T_{ij}\}} \{(u,v) \mid ar(R,u,v) \;\&\; \exists \varrho \in L(\mathfrak{A}_i)\, (u,v) \in \overline{\mathcal{E}}(\varrho)\} \;\cup \\
& \bigcup_{\{i \mid \exists j\, R = inv(T_{ij})\}} \{(u,v) \mid ar(R,u,v) \;\&\; \exists \varrho \in L(\mathfrak{A}_i)\, (v,u) \in \overline{\mathcal{E}}(\varrho)\}, \\
R^{\mathcal{I}} \;=\; & \{(u_0,u_n) \mid \exists u_1, \dots, u_{n-1}\, ((u_i,u_{i+1}) \in \overline{\mathcal{E}}(S_{i+1}) \wedge S_1 S_2 \dots S_n \in L(\mathfrak{A}_R))\}.
\end{aligned}
$$

We need $\overline{\mathcal{E}}(R)$ to adjust $\mathcal{E}(R)$ by taking account of the omitted R-arrows for RIs of the form (C)–(F) as we do not use these RIs in the construction of $\mathfrak{A}_R$. The picture below illustrates such a situation for a role Q and two RIs $QQ \sqsubseteq Q$ and $R \sqsubseteq QP$ ($\mathcal{E}(Q) \subseteq \overline{\mathcal{E}}(Q) \subseteq Q^{\mathcal{I}}$).



We have to show that $\mathcal{I}$ is a model of C_0 and $\mathcal{R}$. To this end, we require the following:

Proposition 20 (i) *If $(u,v) \in \mathcal{E}(R)$ then $ar(R,u,v)$.*
(ii) *If $(u,v) \in \overline{\mathcal{E}}(R)$ then $ar(R,u,v)$.*
(iii) *If $\varrho \in L(\mathfrak{A})$, $(u,v) \in \overline{\mathcal{E}}(\varrho)$ and $\forall \mathfrak{A}^s.\mathfrak{C} \in \mathfrak{a}(u)$ then $\forall \mathfrak{A}^a.\mathfrak{C} \in \mathfrak{a}(v)$.*
(iv) *If $(u,v) \in R^{\mathcal{I}}$ and $\forall \mathfrak{A}^s_R.\mathfrak{C} \in \mathfrak{a}(u)$ then $\forall \mathfrak{A}^a_R.\mathfrak{C} \in \mathfrak{a}(v)$.*

Proof. (i) Follows from (p20) and (p9). More precisely, if $(u,v) \in \mathcal{E}(R)$ then, by (p9), $(v,u) \in \mathcal{E}(inv(R))$. By (p20), $(u,v) \in \mathcal{E}(R)$ implies $\mathfrak{a}(u)|^{\forall}_R \subseteq \mathfrak{a}(v)$, while $(v,u) \in \mathcal{E}(inv(R))$ implies $\mathfrak{a}(v)|^{\forall}_{inv(R)} \subseteq \mathfrak{a}(u)$. Thus, we obtain $ar(R,u,v)$.

(ii) Follows from (i) and the definition of $\overline{\mathcal{E}}(R)$.

(iii) Let $\varrho = S_1 \dots S_n$. Since $(u,v) \in \overline{\mathcal{E}}(\varrho)$, we have $u = u_0, \dots, u_n = v$ with $(u_{i-1},u_i) \in \overline{\mathcal{E}}(S_i)$, for $i = 1, \dots, n$. On other hand, since $S_1 \dots S_n \in L(\mathfrak{A})$, there are $s = p_0, \dots, p_n = a$ such that $p_i \in \delta_{\mathfrak{A}}(p_{i-1}, S_i)$. We have $\forall \mathfrak{A}^{p_0}.\mathfrak{C} \in \mathfrak{a}(u_0)$. If $\forall \mathfrak{A}^{p_i}.\mathfrak{C} \in \mathfrak{a}(u_i)$, $i < n$, then (ii) and $p_{i+1} \in \delta_{\mathfrak{A}}(p_i, S_{i+1})$, $(u_i,u_{i+1}) \in \overline{\mathcal{E}}(S_{i+1})$ give $\forall \mathfrak{A}^{p_{i+1}}.\mathfrak{C} \in \mathfrak{a}(u_{i+1})$. So $\forall \mathfrak{A}^a.\mathfrak{C} \in \mathfrak{a}(v)$.

(iv) Follows from (iii) and the definition of $R^{\mathcal{I}}$. ❑

We show now that $\mathcal{I}$ is a model of $\mathcal{R}$ by considering all types of constraints.

$Dis(S_1,S_2)$: Then the S_i are simple roles, $S_i^{\mathcal{I}} = \mathcal{E}(S_i)$, and so, by (p15), $S_1^{\mathcal{I}} \cap S_2^{\mathcal{I}} = \emptyset$.

$S_1 \sqsubseteq S_2$: The S_i are simple roles and $S_i^{\mathcal{I}} = \mathcal{E}(S_i)$. Thus, if $(u,v) \in S_1^{\mathcal{I}}$ then $(u,v) \in \mathcal{E}(S_1)$ and, by (p16), $(u,v) \in \mathcal{E}(S_2)$; hence $(u,v) \in S_2^{\mathcal{I}}$.

$S_1 \dots S_n \sqsubseteq R$: We have $S_1 \dots S_n \in L(\mathfrak{A}_R)$. If $(u,v) \in (S_1 \dots S_n)^{\mathcal{I}}$ then there are $u = u_0, \dots, u_n = v$ such that $(u_{i-1}, u_i) \in (S_i)^{\mathcal{I}}$, for $i = 1, \dots, n$. By the definition of $(S_i)^{\mathcal{I}}$, there are $u_{i-1} = u_0^i, \dots, u_{n_i}^i = u_i$ with $(u_{j-1}^i, u_j^i) \in \overline{\mathcal{E}}(S_j^i)$, for $1 \leq j \leq n_i$, and $S_1^i S_2^i \dots S_{n_i}^i \in L(\mathfrak{A}_{S_i})$. Therefore, $S_1^1 \dots S_{n_1}^1 S_1^2 \dots S_{n_n}^n \in L(\mathfrak{A}_R)$ and $(u,v) \in R^{\mathcal{I}}$.

$RR \sqsubseteq R$, $RS_1 \dots S_n \sqsubseteq R$ and $S_1 \dots S_n R \sqsubseteq R$ are considered analogously.

$R^- \sqsubseteq R$: As mentioned earlier, each occurrence of R^- is treated as R. It follows that $(u,v) \in \mathcal{E}(R)$ if and only if $(v,u) \in \mathcal{E}(R)$, and $(u,v) \in \overline{\mathcal{E}}(R)$ if and only if $(v,u) \in \overline{\mathcal{E}}(R)$. In addition, $(u,v) \in R^{\mathcal{I}}$ if and only if $(v,u) \in R^{\mathcal{I}}$. Indeed, let $(u,v) \in R^{\mathcal{I}}$. Then, by the definition of $R^{\mathcal{I}}$, there exist $u = u_0, u_1, \dots, u_n = v$ with $(u_i, u_{i+1}) \in \overline{\mathcal{E}}(S_{i+1})$ and $S_1 S_2 \dots S_n \in L(\mathfrak{A}_R)$. Now we have $(u_{i+1}, u_i) \in \overline{\mathcal{E}}(inv(S_{i+1}))$ and, by the construction of $\mathfrak{A}_R$, $inv(S_n) \dots inv(S_1) \in L(\mathfrak{A}_R)$, and so $(v,u) \in R^{\mathcal{I}}$.

$R_i \sqsubseteq Q_i P_{i1} \dots P_{im_i}$: Let $(u,v) \in R_i^{\mathcal{I}}$. Then, by (p19), we have $\forall \mathfrak{A}_i^s.\mathfrak{C} \in \mathfrak{a}(u)$, where s is the initial state of $\mathfrak{A}_i$ and $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \mathfrak{a}(u)) = \exists inv(P_{im_i}). \cdots \exists inv(P_{i1}).(Q_i, \mathfrak{a}(u)|_{Q_i}^{\forall}, \mathfrak{a}(u) \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{inv(Q_i)}^{\forall})$. By Proposition 20, $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(v)$, where a is an accepting state. Now, by (p21), there are $v_0, v_1, \dots, v_{m_i} = v$ such that $(v_j, v_{j-1}) \in \mathcal{E}(inv(P_{ij}))$, $\mathfrak{a}(u)|_{Q_i}^{\forall} \subseteq \mathfrak{a}(v_0)$ and $\mathfrak{a}(v_0)|_{inv(Q_i)}^{\forall} \subseteq \mathfrak{a}(u) \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{inv(Q_i)}^{\forall} \subseteq \mathfrak{a}(u)$, that is, $(v_0, v) \in (P_{i1} \dots P_{im_i})^{\mathcal{I}}$ and $ar(Q_i, u, v_0)$. Hence, $(u, v_0) \in Q_i^{\mathcal{I}}$ and $(u,v) \in (Q_i P_{i1} \dots P_{im_i})^{\mathcal{I}}$.

$R_i \sqsubseteq T_{i1} \sqcup \dots \sqcup T_{im_i}$: Let $(u,v) \in R_i^{\mathcal{I}}$. Then, by (p19), we have $\forall \mathfrak{A}_i^s.\mathfrak{C} \in \mathfrak{a}(u)$, where s is the initial state of $\mathfrak{A}_i$ and $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \mathfrak{a}(u)) = \bigvee_{h=1}^{m_i} (T_{ih}, \mathfrak{a}(u)|_{T_{ih}}^{\forall}, \mathfrak{a}(u) \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{inv(T_{ih})}^{\forall})$. By Proposition 20, $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(v)$, where a is an accepting state. Now, by (p22), there is $j \in \{1, \dots, m_i\}$ such that $\mathfrak{a}(u)|_{T_{ij}}^{\forall} \subseteq \mathfrak{a}(v)$ and $\mathfrak{a}(v)|_{inv(T_{ij})}^{\forall} \subseteq \mathfrak{a}(u) \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{inv(T_{ij})}^{\forall} \subseteq \mathfrak{a}(u)$, i.e., $ar(T_{ij}, u, v)$. Hence, $(u,v) \in T_{ij}^{\mathcal{I}}$ and $(u,v) \in (T_{i1} \sqcup \dots \sqcup T_{im_i})^{\mathcal{I}}$.

$R_i = Q_i P_{i1} \dots P_{im_i}$: Let $(u,v) \in R_i^{\mathcal{I}}$. Then there exists a role chain $\varrho \in L(\mathfrak{A}_{R_i})$ such that $(u,v) \in \overline{\mathcal{E}}(\varrho)$. If $\varrho \in L(\mathfrak{A}_i)$ then $(u,v) \in (Q_i P_{i1} \dots P_{im_i})^{\mathcal{I}}$, and the proof is same as for $R_i \sqsubseteq Q_i P_{i1} \dots P_{im_i}$. So suppose $\varrho \notin L(\mathfrak{A}_i)$ is shortest possible. Since $\varrho \sqsubseteq_{\mathcal{R}'}^* R_i$, there is a sequence $\varrho \sqsubseteq_{\boldsymbol{r}_1} \varrho_1 \sqsubseteq_{\boldsymbol{r}_2} \cdots \sqsubseteq_{\boldsymbol{r}_n} \varrho_n \sqsubseteq_{\boldsymbol{r}_{n+1}} R_i$, where $\boldsymbol{r}_j \in \mathcal{R}'$, for $1 \leq j \leq n+1$, and at least one $\boldsymbol{r}_j$ is not in $\mathcal{R}^i$. If $\boldsymbol{r}_j \notin \mathcal{R}^i$, for $j < n+1$, then we can find a shorter ϱ, and so $\boldsymbol{r}_{n+1} = (Q_i P_{i1} \dots P_{im_i} \sqsubseteq R_i)$. Therefore, $\varrho = \varrho_0' \varrho_1' \dots \varrho_{m_i}'$ is such that $\varrho_0' \sqsubseteq_{\mathcal{R}'}^* Q_i$ and $\varrho_j' \sqsubseteq_{\mathcal{R}'}^* P_{ij}$, for $1 \leq j \leq m_i$. Thus, $(u,v) \in (Q_i P_{i1} \dots P_{im_i})^{\mathcal{I}}$.

Let now $(u,v) \in (Q_i P_{i1} \dots P_{im_i})^{\mathcal{I}}$. Then there exists v_0 such that $(u, v_0) \in Q_i^{\mathcal{I}}$ and $(v_0, v) \in (P_{i1} \dots P_{im_i})^{\mathcal{I}}$. In the case $(u, v_0) \in \overline{\mathcal{E}}(Q_i) \backslash \mathcal{E}(Q_i)$ then by construction of $\overline{\mathcal{E}}(Q_i)$ we have $(\exists \varrho \in L(\mathfrak{A}_i))(\exists z)\, (u,z) \in \overline{\mathcal{E}}(\varrho)$. If $v = z$ then we have $(u,v) \in R_i^{\mathcal{I}}$. Otherwise the proof is same as for $Q_i P_{i1} \dots P_{im_i} \sqsubseteq R_i$.

$R_i = T_{i1} \sqcup \dots \sqcup T_{im_i}$: Similar to the previous case.

To prove that $\mathcal{I}$ satisfies C_0, we show that

$$C \in \mathfrak{c}(u) \text{ implies } u \in C^{\mathcal{I}}, \text{ for each } u \in \boldsymbol{S} \text{ and each } C \in con(C_0). \tag{14}$$

Together with (p1), this will imply $u_0 \in (C_0)^{\mathcal{I}}$. We prove (14) by induction on the construction of concepts. If C is a concept name then (14) follows from the definition. For $\bot$ and $\top$, it follows from (p3), and for $C_1 \sqcap C_2$, $C_1 \sqcup C_2$, $\exists R.C$, $\geq qS.C$, and $\exists S.\mathit{Self}$, from (p6), (p7), (p8), (p11) and (p4). The case of $\neg C$ follows from (p2) and (p5) and case of $\leq qS.C$ follows from (p10) and (p12). Consider now the (only interesting) case $C \equiv \forall R.D$. Let $\forall R.D \in \mathfrak{c}(u)$ and $(u,v) \in R^{\mathcal{I}}$. By (p17) we have $\forall \mathfrak{A}^s_R.D \in \mathfrak{a}(u)$, where s is the initial state. Therefore, by Proposition 20, we have $\forall \mathfrak{A}^a_R.D \in \mathfrak{a}(v)$ and a is an accepting state. Now, by (p18), $D \in \mathfrak{c}(v)$; by IH, $v \in D^{\mathcal{I}}$, and thus $u \in (\forall R.D)^{\mathcal{I}}$.

For $o \in \mathit{nom}(C_0)$, by (p14), there is v_o with $o \in \mathfrak{c}(v_o)$, and so $v_o \in o^{\mathcal{I}}$. If $u \in o^{\mathcal{I}}$ then $o \in \mathfrak{c}(u)$, and so, by (p13), $u = v_o$. Thus, $o^{\mathcal{I}}$ is a singleton set.

($\Rightarrow$) Suppose $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of C_0 and $\mathcal{R}$. We define $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}, \mathfrak{a}, \mathcal{E})$ by taking

$$\boldsymbol{S} = \Delta^{\mathcal{I}}, \quad \mathcal{E}(R) = R^{\mathcal{I}}, \quad \mathfrak{c}(u) = \{C \in \mathit{con}(C_0) \mid u \in C^{\mathcal{I}}\} \cup \{\top\}$$

and define $\mathfrak{a}(u)$ as follows. First, we define by induction on $\lhd$ auxiliary sets $\mathfrak{a}'(u, \boldsymbol{r})$, where $\boldsymbol{r}$ is an RI of the from (C)–(F). Recalling that $\boldsymbol{r}_0 \lhd \boldsymbol{r}_i$ for all i, $1 \leq i \leq l$, we set

$$\begin{aligned}\mathfrak{a}'(u, \boldsymbol{r}_0) \;=\; & \{\forall \mathfrak{A}^s_R.C \mid s \text{ is the initial state}, \forall R.C \in \mathit{con}(C_0) \text{ and } u \in (\forall R.C)^{\mathcal{I}}\} \;\cup \\ & \{\forall \mathfrak{A}^q_R.C \in \boldsymbol{qc} \mid \text{ for all } S_1 S_2 \ldots S_n \in L(\mathfrak{A}^q_R), u \in (\forall S_1.\forall S_2.\cdots\forall S_n.C)^{\mathcal{I}} \\ & \qquad\qquad \text{and if } \varepsilon \in L(\mathfrak{A}^q_R) \text{ then } u \in C^{\mathcal{I}}\}.\end{aligned}$$

Then, assuming that $\mathfrak{a}'(u, \boldsymbol{r}')$ is defined for every $\boldsymbol{r}' \lhd \boldsymbol{r_i}$, we set

$$\begin{aligned}\mathfrak{a}'(u, \boldsymbol{r}_i) = \{\forall \mathfrak{A}^q_i.\mathfrak{C} \mid \exists v \in \Delta^{\mathcal{I}}\, \exists w \in (\mathit{role}(C_0, \mathcal{R}))^* \; (w, q) \in \mathit{prefix}\, L(\mathfrak{A}_i), \\ (v, u) \in (w)^{\mathcal{I}}, \; \mathfrak{C} = \Xi(\boldsymbol{r}_i, \bigcup_{\boldsymbol{r}' \lhd \boldsymbol{r_i}} \mathfrak{a}'(v, \boldsymbol{r}'))\},\end{aligned}$$

where $(w)^{\mathcal{I}} = S^{\mathcal{I}}_1 \ldots S^{\mathcal{I}}_n$, for $w = S_1 \ldots S_n$, and

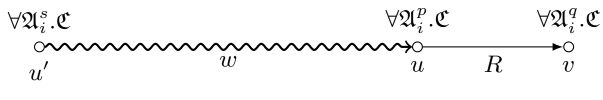
$$\mathit{prefix}\, L(\mathfrak{A}_i) = \{(w, q) \mid q \text{ a state in } \mathfrak{A}_i, \; \forall w' \in L(\mathfrak{A}^q_i) \; ww' \in L(\mathfrak{A}_i)\}.$$

Note that $\{\forall \mathfrak{A}^s_i.\mathfrak{C} \mid s$ the initial state, $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \bigcup_{\boldsymbol{r}' \lhd \boldsymbol{r_i}} \mathfrak{a}'(u, \boldsymbol{r}')\} \subseteq \mathfrak{a}'(u, \boldsymbol{r}_i)$.

Finally, we set

$$\mathfrak{a}(u) \;=\; \bigcup_{j=0}^{l} \mathfrak{a}'(u, \boldsymbol{r}_j).$$

We now prove that $\boldsymbol{T}$ is a tableau for C_0 w.r.t. $\mathcal{R}$. Properties (p1)–(p16) follow immediately from the definitions of $\mathfrak{c}$ and $\mathcal{E}$, while (p17)–(p19) follow from the definitions of $\mathfrak{c}(u)$ and $\mathfrak{a}(u)$. For (p20), suppose $(u, v) \in \mathcal{E}(R)$, $\forall \mathfrak{A}^p.\mathfrak{C} \in \mathfrak{a}(u)$ and $q \in \delta_{\mathfrak{A}}(p, R)$. Then $\forall \mathfrak{A}^p.\mathfrak{C} \in \mathfrak{a}'(u, \boldsymbol{r}_i)$, for some i. If $i > 0$ then $\mathfrak{A} = \mathfrak{A}_i$ and, by the definition of $\mathfrak{a}'(u, \boldsymbol{r}_i)$, there are $u' \in \Delta^{\mathcal{I}}$ and $w \in (\mathit{role}(C_0, \mathcal{R}))^*$ with $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \bigcup_{\boldsymbol{r}' \lhd \boldsymbol{r_i}} \mathfrak{a}'(u', \boldsymbol{r}'))$, $(w, p) \in \mathit{prefix}\, L(\mathfrak{A})$ and $(u', u) \in (w)^{\mathcal{I}}$. Let $w' = wR$. Then $(w', q) \in \mathit{prefix}\, L(\mathfrak{A})$ and $(u', v) \in (w')^{\mathcal{I}}$, so $\forall \mathfrak{A}^q.\mathfrak{C} \in \mathfrak{a}'(v, \boldsymbol{r}_i) \subseteq \mathfrak{a}(v)$.



For $i = 0$ (i.e., when $\mathfrak{C}$ is a concept C and $\forall\mathfrak{A}^p.C \in \mathfrak{a}'(u, \boldsymbol{r}_0)$), suppose $\forall\mathfrak{A}^q.C \notin \mathfrak{a}'(v, \boldsymbol{r}_0)$. By the definition of $\mathfrak{a}'(v, \boldsymbol{r}_0)$, this can be for two reasons (Horrocks et al., 2006):

– There is $S_2 \dots S_n \in L(\mathfrak{A}^q)$ and $v \notin (\forall S_2. \cdots \forall S_n.C)^{\mathcal{I}}$. However, this implies that $RS_2 \dots S_n \in L(\mathfrak{A}^p)$ and $u \notin (\forall R.\forall S_2. \cdots \forall S_n.C)^{\mathcal{I}}$, contrary to $\forall\mathfrak{A}^p.C \in \mathfrak{a}'(u)$.

– $\varepsilon \in L(\mathfrak{A}^q)$ and $v \notin C^{\mathcal{I}}$. But then $R \in L(\mathfrak{A}^p)$ and $u \notin (\forall R.C)^{\mathcal{I}}$, which is again a contradiction.

Therefore, $\forall\mathfrak{A}^q.C \in \mathfrak{a}'(v, \boldsymbol{r}_0)$, and so $\mathfrak{a}(u)|_R^{\forall} \subseteq \mathfrak{a}(v)$.

To show (p21) and (p22), suppose $\forall\mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(u)$, where a is an accepting state. By the definition of $\mathfrak{a}(u)$, there are $v \in \Delta^{\mathcal{I}}$ and $w \in (\mathit{role}(C_0, \mathcal{R}))^*$ such that $(w, a) \in \mathit{prefix}\, L(\mathfrak{A}_i)$, $(v, u) \in (w)^{\mathcal{I}}$ and $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \bigcup_{\boldsymbol{r}' \lhd \boldsymbol{r}_i} \mathfrak{a}'(v, \boldsymbol{r}')) = \Xi(\boldsymbol{r}_i, \mathfrak{a}(v))$. Since a is an accepting state, we have $w \in L(\mathfrak{A}_i)$, and so $(v, u) \in (R_i)^{\mathcal{I}}$.

For (p21)—i.e., $i \leq k$—we have $\mathfrak{C} = \exists \mathit{inv}(P_{im_i}). \cdots \exists \mathit{inv}(P_{i1}).(t^r, t^{\forall}, t^-)$, where $t^r = Q_i$, $t^{\forall} = \mathfrak{a}(v)|_{Q_i}^{\forall}$, $t^- = \mathfrak{a}(v) \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{\mathit{inv}(Q_i)}^{\forall}$. We also have $(v, u) \in (Q_i P_{i1} \dots P_{im_i})^{\mathcal{I}}$, and there are $v_0, v_1, \dots, v_{m_i} = u$ such that $(v_{j-1}, v_j) \in P_{ij}^{\mathcal{I}}$ and $(v, v_0) \in Q_i^{\mathcal{I}}$. Therefore, by (p20), $t^{\forall} \subseteq \mathfrak{a}(v_0)$ and $\mathfrak{a}(v_0)|_{\mathit{inv}(t^r)}^{\forall} \subseteq t^-$.

For (p22)—i.e., $i > k$—we have $\mathfrak{C} = \bigvee_{h=1}^{m_i}(t_h^r, t_h^{\forall}, t_h^-)$, where $t_h^r = T_{ih}$, $t_h^{\forall} = \mathfrak{a}(v)|_{T_{ih}}^{\forall}$, $t_h^- = \mathfrak{a}(v) \cap \boldsymbol{qc}(C_0, \mathcal{R})|_{\mathit{inv}(T_{ih})}^{\forall}$ for $1 \leq h \leq m_i$. We also have $(v, u) \in (T_{i1} \sqcup \cdots \sqcup T_{im_i})^{\mathcal{I}}$, and there is $j \in \{1, \dots, m_i\}$ such that $(v, u) \in T_{ij}^{\mathcal{I}}$. Therefore, by (p20), $t_j^{\forall} \subseteq \mathfrak{a}(u)$ and $\mathfrak{a}(u)|_{\mathit{inv}(t_j^r)}^{\forall} \subseteq t_j^-$.

(p23) is considered in the same way as (p20). ❑

Appendix B. The Tableau Algorithm

The tableau algorithm, for $\mathcal{SR}^+\mathcal{OIQ}$ concepts C_0 and RBoxes $\mathcal{R}$, works on completion graphs similarly to the algorithms given by Horrocks et al. (2006) and Horrocks and Sattler (2007). To present it, we require some additional notation. We assume that the given $\mathcal{R}$ is same as in Appendix A with $\mathcal{R}_{C,D,E,F} = \{\boldsymbol{r}_i \mid i = 1, \dots, l\}$. For $1 \leq i \leq l$ and a basic role P, where $P = Q_i$ for $i \leq k$, and $P \in \{T_{i1}, \dots, T_{im_i}\}$ for $i > k$, let

$$\boldsymbol{qc}^*(\boldsymbol{r}_i, P) = \{\forall\mathfrak{A}^p.\mathfrak{C} \mid \text{there exists } q \in \delta_{\mathfrak{A}}(p, P),\ \forall\mathfrak{A}^p.\mathfrak{C} \in \bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)\} \cup \{\forall\mathfrak{A}^q.\mathfrak{C} \mid q \in \delta_{\mathfrak{A}}(p, \mathit{inv}(P)),\ \forall\mathfrak{A}^p.\mathfrak{C} \in \bigcup_{\boldsymbol{r}_j \lhd \boldsymbol{r}_i} \boldsymbol{qc}(\boldsymbol{r}_j)\}.$$

We now set $\boldsymbol{qc}^*(\boldsymbol{r}_i) = \boldsymbol{qc}^*(\boldsymbol{r}_i, Q_i)$, for $i \leq k$, and $\boldsymbol{qc}^*(\boldsymbol{r}_i) = \bigcup_{j=1}^{m_i} \boldsymbol{qc}^*(\boldsymbol{r}_i, T_{ij})$, for $i > k$. 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:

– $\boldsymbol{qc}(C_0, \mathcal{R}) \cup \{\neg\forall\mathfrak{A}^p.\mathfrak{C} \mid \forall\mathfrak{A}^p.\mathfrak{C} \in \bigcup_{i=1}^{l} \boldsymbol{qc}^*(\boldsymbol{r}_i)\} \subseteq \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$,

– if $\forall\mathfrak{A}^p.\mathfrak{C} \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$ 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 $\mathfrak{C} \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$,

– if $(\bigvee_{j=1}^{m} \mathfrak{C}_j) \in \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$ then $\{\mathfrak{C}_1, \dots, \mathfrak{C}_m\} \subseteq \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$.

Unlike $\boldsymbol{qc}(C_0, \mathcal{R})$, the set $\overline{\boldsymbol{qc}}(C_0, \mathcal{R})$ contains sub-quasi-concepts. (Quasi-concepts of the form $\neg\forall\mathfrak{A}^p.\mathfrak{C}$ will only be used to make sure that $\forall\mathfrak{A}^p.\mathfrak{C}$ does not belong to a label.)

Given an $\mathcal{SR}^+\mathcal{OIQ}$ concept C_0 and an RBox $\mathcal{R}$, a *completion graph for* C_0 and $\mathcal{R}$ is a structure of the form $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$, where

– $V_1 \cap V_2 = \emptyset$; the elements of V_1 are called *root nodes*, and the elements of V_2 are called *internal* (or *non-root*) *nodes*;

– (V, E_1) is a directed forest with nodes $V = V_1 \cup V_2$ and arcs E_1 (its roots have no incoming arcs);

– E_2 is a set of arcs between nodes and root nodes, as well as arcs of the form (x, x), for $x \in V_2$;

– for each $(x, y) \in E$, where $E = E_1 \cup E_2$, we have $\mathfrak{l}(x, y) \subseteq \mathit{role}(C_0, \mathcal{R})$; if $R' \in \mathfrak{l}(x, y)$ and $R' \sqsubseteq^* R$, 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 $\mathit{inv}(R)$-successor of y; also, x is called an ε-neighbour of x (cf. Horrocks et al., 2006);

– for each $x \in V$, $\mathfrak{c}(x) \subseteq \mathit{con}(C_0) \cup \{\leq mS.C \mid \leq nS.C \in \mathit{con}(C_0)$, and $m < n\}$ and $\mathfrak{a}(x) \subseteq \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$;

– $\not\cong$ is a symmetric binary relation on V;

– for each $o \in \mathit{nom}(C_0)$, there is $x \in V_1$ such that $o \in \mathfrak{c}(x)$.

Following Horrocks et al. (2006) and Horrocks and Sattler (2007), we distinguish between two sets of nodes: those in V_1 can be arbitrarily interconnected (they are called root nodes), while those in V_2 form a tree structure (they are called internal nodes). Intuitively, a completion graph is a collection of trees whose root nodes can be arbitrarily connected and there may also be arcs from internal nodes to root nodes (see Fig. 2 on page 841). We also distinguish between two sets of arcs: those in E_1 connect nodes in the same tree, while those in E_2 are the remaining arcs in the graph.

To illustrate the difference between R-successors and neighbours, suppose $(R' \sqsubseteq R) \in \mathcal{R}$

$$x \xrightarrow{\qquad \mathfrak{l}(x,y) = \{R'\} \qquad} y$$

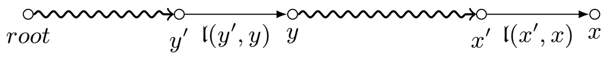
and $\mathfrak{l}(x, y) = \{R'\}$, as in the picture above. Then y is both an R'- and R-successor of x, but x is neither an $\mathit{inv}(R')$- nor an $\mathit{inv}(R)$-successor of y; y is both an R'- and R-neighbour of x, and x is an $\mathit{inv}(R')$- and $\mathit{inv}(R)$-neighbour of y.

To ensure that the tableau algorithm eventually comes to a stop, we use a blocking technique that is similar to the one of Horrocks et al. (2006). A node $x \in V_2$ is called *blocked* if it is either directly or indirectly blocked. A node $x \in V_2$ is *directly blocked* if none of its (not necessarily immediate) E_1-ancestors is blocked, and there are nodes x', y and y' such that:

– y' is not a root,

- $(x', x) \in E_1$, $(y', y) \in E_1$ and y is an E_1-ancestor of x',
- $\mathfrak{c}(x) = \mathfrak{c}(y)$, $\mathfrak{c}(x') = \mathfrak{c}(y')$, $\mathfrak{a}(x) = \mathfrak{a}(y)$, $\mathfrak{a}(x') = \mathfrak{a}(y')$ 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_1-ancestors is blocked.

For a simple role S, $x \in V$ and $C \in con(C_0)$, let

$$S^{\boldsymbol{G}}(x, C) = \{y \mid y \text{ is an } S\text{-neighbour of } x,\ C \in \mathfrak{c}(y) \text{ and if } x \in V_1 \text{ then } y \text{ is not indirectly blocked}\}.$$

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

- $\bot \in \mathfrak{c}(x)$,
- $\{A, \neg A\} \subseteq \mathfrak{c}(x)$, for a concept name A,
- $\{\forall \mathfrak{A}^p.\mathfrak{C}, \neg\forall \mathfrak{A}^p.\mathfrak{C}\} \subseteq \mathfrak{a}(x)$, for a quasi-concept $\forall \mathfrak{A}^p.\mathfrak{C} \in \bigcup_{i=1}^{l} \boldsymbol{qc}^*(\boldsymbol{r}_i)$,
- x is an S-neighbour of x and $\neg\exists S.Self \in \mathfrak{c}(x)$,
- $Dis(R, S) \in \mathcal{R}$, while y is both an R- and an S-neighbour of x, for some $y \in V$,
- $(\leq nS.C) \in \mathfrak{c}(x)$, while $\{y_0, \dots, y_n\} \subseteq S^{\boldsymbol{G}}(x, C)$ with $y_i \not\cong y_j$, for $0 \leq i < j \leq n$,
- for some $o \in nom(C_0)$, there is node $y \not\cong x$ with $o \in \mathfrak{c}(x) \cap \mathfrak{c}(y)$,
- $\mathfrak{C} = (t^r, t^\forall, t^-) \in \mathfrak{a}(x)$ and $\mathfrak{a}(x)|^{\forall}_{inv(t^r)} \not\subseteq t^-$.

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

To simplify the tableau rules, we require some terminology and notation originally used by Horrocks et al. (2006) and Horrocks and Sattler (2007). An R-neighbour y of x is said to be *safe* if either $x \in V_2$ or $x \in V_1$ and y is not blocked. The result (and the procedure) of *pruning* a node y in $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$, denoted $Prune(y)$, is the graph obtained from $\boldsymbol{G}$ in the following way: we remove every (y, z) from E and, if $z \in V_2$, $Prune(z)$; we also remove y from V. The result (and the procedure) of *merging* nodes y and x in $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$, denoted $Merge(y, x)$, is the graph obtained from $\boldsymbol{G}$ as follows:

1. for all z such that $(z, y) \in E$:
   - if $\{(x, z), (z, x)\} \cap E = \emptyset$, then add (z, x) to E (to E_1 if $x \in V_2$, otherwise to E_2) and set $\mathfrak{l}(z, x) := \mathfrak{l}(z, y)$,
   - if $(z, x) \in E$, then set $\mathfrak{l}(z, x) := \mathfrak{l}(z, x) \cup \mathfrak{l}(z, y)$,
   - if $(x, z) \in E$, then set $\mathfrak{l}(x, z) := \mathfrak{l}(x, z) \cup \{inv(R) \mid R \in \mathfrak{l}(z, y)\}$, and

– remove (z, y) from E;

2. for all root nodes z such that $(y, z) \in E_2$:

   – if $\{(x, z), (z, x)\} \cap E = \emptyset$, then add (x, z) to E_2 and set $\mathfrak{l}(x, z) := \mathfrak{l}(y, z)$,
   – if $(x, z) \in E$, then set $\mathfrak{l}(x, z) := \mathfrak{l}(x, z) \cup \mathfrak{l}(y, z)$,
   – if $(z, x) \in E$, then set $\mathfrak{l}(z, x) := \mathfrak{l}(z, x) \cup \{inv(R) \mid R \in \mathfrak{l}(y, z)\}$, and
   – remove (y, z) from E_2;

3. set $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \mathfrak{c}(y)$ and $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \mathfrak{a}(y)$;

4. add $x \not\cong z$, for all z with $y \not\cong z$;

5. $Prune(y)$.

Let $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$ be a completion graph. The completion rules can extend $\boldsymbol{G}$ in two ways: by adding a new leaf and by adding a new root. We say that a node $x \in V_2$, with $(y, x) \in E_1$, is of *level* i in the forest (V, E_1) if either $i = 1$ and $y \in V_1$, or $i > 1$ and $y \in V_2$ is of level $i-1$ in (V, E_1). A node $x \in V_1$ is of *level* i in the graph $(V_1, E_2 \cap (V_1 \times V_1))$ if either $i = 0$ and there exists $o \in nom(C_0)$ such that $o \in \mathfrak{c}(x)$, or $i > 0$, x is not of level $\leq (i-1)$ in $(V_1, E_2 \cap (V_1 \times V_1))$ and there is $y \in V_1$ of level $i - 1$ in $(V_1, E_2 \cap (V_1 \times V_1))$ with $(y, x) \in E_2$.

The tableau rules will be applied according to the following strategy: the (o)-rule is of highest priority; after that we apply the $(=_r)$- and $(\leq_r)$-rules, starting with root nodes of lower levels; applications of all other rules follow.

Our tableau algorithm is non-deterministic. It takes a $\mathcal{SR}^+\mathcal{OIQ}$ 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. The algorithm starts by constructing the completion graph $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$, where

– $V_1 = \{x_o \mid o \in nom(C_0)\} \cup \{x_{C_0}\}$,

– $V_2 = \emptyset$,

– $E_1 = \emptyset$, $E_2 = \emptyset$,

– $\mathfrak{c}(x_o) = \{o\}$, $\mathfrak{c}(x_{C_0}) = \{C_0\}$,

– $\mathfrak{a}(x_o) = \emptyset$, $\mathfrak{a}(x_{C_0}) = \emptyset$,

– $\mathfrak{l} = \emptyset$,

– $\not\cong$ is empty.

Then the algorithm non-deterministically applies one of the completion rules given in Tables 1 and 2; 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'.

To prove that this algorithm always comes to a stop and returns a correct answer, we require the following lemma:

Lemma 21 *Let $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$ be the structure constructed at some step of the algorithm. Then, for every $x \in V_2$, there exists exactly one $y \in V$ such that $(y, x) \in E_1$.*

Proof. The proof is by induction on the number of steps. The basis of induction ($V_2 = \emptyset$) is trivial. So suppose that our claim holds for some step and consider what happens after an application of a completion rule. By applying the rules ($\exists$), (r8) and ($\geq$) to a node x, we add one or more nodes to V_2, with x being the only predecessor of these nodes. Also, we have to consider the rules ($\leq$), (o) and ($=_r$), which merge nodes, because other rules do not change V_2 and E_1. Merging nodes changes the graph by possibly adding new edges, deleting some edges and pruning some nodes. Observe that if we prune a node y then our claim still holds because we delete the successors of y which belong to V_2. Deleting an edge does not spoil the claim either. Thus, it is enough to examine the cases when a new edge is added to E_1. If $\mathit{Merge}(y, x)$ and $x \in V_1$ then all newly added edges belong to E_2, and so after applying $\mathit{Merge}(y, x)$ the claim still holds. The only interesting case is when we apply $\mathit{Merge}(z, y)$ in the rule ($\leq$) with $(\leq nS.C) \in \mathfrak{c}(x)$, $y, z \in S^{\boldsymbol{G}}(x, C)$ and $y, z \in V_2$. Because of $y, z \in V_2$, the nodes y and z have only one parent node, although z is not an ancestor of y; so the following fours cases are possible.

Case 1: $(y, x) \in E_1$ and $(x, z) \in E_1$. Then no new edge is added to E_1, and the claim holds.

Case 2: $(x, y) \in E_1$ and $(x, z) \in E_1$. Again, we do not add a new edge to E_1.

Case 3: $(y, x) \in E_1$ and $(z, x) \in E_1$. In this case $x \in V_2$ and x has two parent nodes y and z, which is impossible by IH.

Case 4: $(y, x) \in E_2$ or $(z, x) \in E_2$. This case is not possible either because $x \in V_1$, $(\leq nS.C) \in \mathfrak{c}(x)$, y (or z) is an S-neighbour of x, and so before applying ($\leq$) we have to apply ($\leq_r$) or ($=_r$) in view of their higher priority. ❑

We can now show termination.

Lemma 22 *The tableau algorithm always terminates.*

Proof. The sets $\mathit{con}(C_0)$, $\overline{\boldsymbol{qc}}(C_0, \mathcal{R})$, $\mathit{role}(C_0, \mathcal{R})$ we use in the labels of nodes and edges are finite. Let $l_0 = \sharp \mathit{nom}(C_0)$, $l_1 = \sharp \mathit{con}(C_0)$, $l_2 = \sharp \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$, $l_3 = \sharp \mathit{role}(C_0, \mathcal{R})$ and $n_{\max} = \max\{n \mid (\geq nR.C) \in \mathit{con}(C_0) \text{ or } (\leq nR.C) \in \mathit{con}(C_0)\}$. The completion graph and the completion rules have following properties. Each node x is labelled with two sets $\mathfrak{c}(x) \subseteq \mathit{con}(C_0)$ and $\mathfrak{a}(x) \subseteq \overline{\boldsymbol{qc}}(C_0, \mathcal{R})$. The number of different pairs of such labels does not exceed $2^{l_1+l_2}$. Each edge (x, y) is labelled with a set $\mathfrak{l}(x, y) \subseteq \mathit{role}(C_0, \mathcal{R})$, so the number of different labels of edges is at most 2^{l_3}. The number of different labels for a pair of nodes connected by an arc is at most $L = 2^{l_3+2l_1+2l_2}$. Therefore, any path in the forest (V, E_1), which starts from a root node and is of length $\geq L + 2$, contains a blocked node. Every application of any rule is determined by some (quasi-) concept and node, with the same rule applicable to the same (quasi-) concept and node only once. The completion rules never remove labels from nodes in the graph, and the only rules that remove nodes are ($\leq$), (o) and ($=_r$). Only ($\exists$), ($\geq$), ($\leq_r$) and (r8) generate new nodes, and each such generation is triggered by a (quasi-) concept of the form $\exists R.C$, $\geq nR.C$, $\leq nR.C$ or $\exists P.\mathfrak{C}$ in the label of a node x. The number of the concepts is $\leq l_1 + l_2$. The rules ($\geq$) and ($\leq_r$) can generate at most $n_{\max}$ successors of a given node, for each concept of the form $\geq nR.C$ or $\leq nR.C$. The other two rules generate only one successor for each concept. It follows that the number of created outgoing arcs for a node does not exceed $l_1 \cdot n_{max} + l_2$. If a node y is removed from $\boldsymbol{G}$ by $(\leq)$, (o) or $(=_r)$, its label migrates to the node z. So the rules $(\exists)$, $(\geq)$, $(\leq_r)$ and (r8), which generate y that is later merged by $(\leq)$, (o) or $(=_r)$, will not be applied again to the same node.

Now, we show that the number of nodes in the completion graph is limited. Together with the observations above, this will mean that we can apply the completion rules finitely many times, and so the algorithm will eventually come to a stop.

To this end, we require the following claim: if $x \in V_1$ is of level i in $(V_1, E_2 \cap (V_1 \times V_1))$, $y \in V_2$ is not indirectly blocked and $(y, x) \in E_2$, then y is of level $\leq L + 2 - i$ in (V, E_1). Indeed, for $i = 0$, y is of level $\leq L + 2$ because y is not indirectly blocked and each node of $> L + 2$ level is indirectly blocked. If x is of level $i > 0$ then the only way to add an edge $(y, x) \in E_2$ is first to apply the rule (o), which will add an E_2-edge between some $y_0 \in V_2$ and $x_0 \in V_1$, and then repeatedly apply $(=_r)$. The node x_0 is of level 0 in $(V_1, E_2 \cap (V_1 \times V_1))$, while y_0 is of level $\leq L + 2$ in (V, E_1). If we apply the rule $(=_r)$, then y_0 will be merged with some successor $x_1 \in V_1$ of x_0 created by an application of $(\leq_r)$. The node x_1 is of level ≤ 1, and we add the edge $(y_1, x_1) \in E_2$, where y_1 is a parent of y_0 and of level $\leq L + 2 - 1$ in (V, E_1); see Fig. 2. By repeating the same argument, we see that the node y is of level $\leq L + 2 - i$ in (V, E_1).

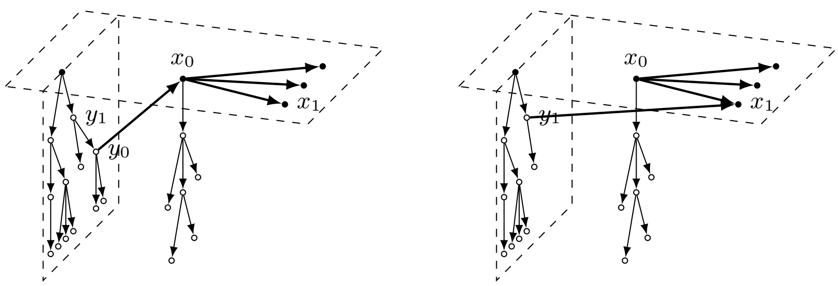


Figure 2: Before and after an application of $(=_r)$; the bold arcs are in E_2 and the nodes $\bullet$ are in V_1.

The claim proved above means that if a node $x \in V_1$ is of level $L+2$ in $(V_1, E_2 \cap (V_1 \times V_1))$, then there is no $y \in V_2$ with $(y, x) \in E_2$. Hence, the rule $(\leq_r)$ cannot be applied to a node x of level $L + 2$, and so there is no root node of level $L + 3$.

The only rule that can add new nodes to V_1 is $(\leq_r)$. It can be applied at most l_1 times to a given node and add at most $l_1 \cdot n_{max}$ successors. At the beginning $V_1 \backslash \{x_{C_0}\}$ contains l_0 nodes, so $(\leq_r)$ can create at most $l_0 \cdot l_1 \cdot n_{\max}$ successors of these nodes. By applying $(\leq_r)$ to them again, we obtain $\leq l_0 \cdot (l_1 \cdot n_{\max})^2$ new nodes (of level ≤ 2). It follows that

$$\sharp V_1 \;\leq\; 1 + \sum_{i=0}^{L+2} l_0 \cdot (l_1 \cdot n_{\max})^i = O(l_0 \cdot (l_1 \cdot n_{\max})^{L+3}).$$

The number of nodes in V_2 is also limited. At the beginning $V_2 = \emptyset$. For each node in V_1, the algorithm can create $\leq l_1 \cdot n_{max} + l_2$ arcs that lead to nodes in V_2. Thus, the number of nodes of level 1 in V_2 does not exceed $\sharp V_1 \cdot (l_1 \cdot n_{max} + l_2)$; the number of their successors is at most $\sharp V_1 \cdot (l_1 \cdot n_{max} + l_2)^2$; and finally,

$$\sharp V_2 \;\leq\; \sum_{i=1}^{L+2} \sharp V_1 \cdot (l_1 \cdot n_{max} + l_2)^i = O(\sharp V_1 \cdot (l_1 \cdot n_{max} + l_2)^{L+3}).$$

This completes the proof of the lemma. ❑

The next lemma shows that the answers returned by the algorithm are correct.

Lemma 23 *The tableau algorithm returns 'yes' if and only if there exists a tableau for C_0 w.r.t. $\mathcal{R}$.*

Proof. ($\Rightarrow$) Suppose the algorithm returns 'yes' by generating a clash-free completion graph $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$ to which no completion rule is applicable. Let $V = V_1 \cup V_2$ and $E = E_1 \cup E_2$. We write $\beta(x) = x$, if $x \in V_1$ or $x \in V_2$ is not blocked; and $\beta(x) = y$, if $x \in V_2$ and y blocks x.

Define a set $\mathit{paths}(\boldsymbol{G})$ inductively by taking (cf. Horrocks et al., 2006):

– if $x_0 \in V_1$ then $(x_0, x_0) \in \mathit{paths}(\boldsymbol{G})$; in this case we write $\mathit{Root}(x_0) = (x_0, x_0)$,

– if $\pi \in \mathit{paths}(\boldsymbol{G})$, a node $z \in V_2$ is not indirectly blocked and $(\mathit{tail}(\pi), z) \in E_1$, then the sequence $\pi, (\beta(z), z)$ is in $\mathit{paths}(\boldsymbol{G})$.

Here $\mathit{tail}(\pi) = x_n$ and $\mathit{tail}'(\pi) = x'_n$, for $\pi = (x_0, x'_0), \dots, (x_n, x'_n)$. The members π of $\mathit{paths}(\boldsymbol{G})$ will be called *paths* in $\boldsymbol{G}$.

We now define a tableau $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}', \mathfrak{a}', \mathcal{E})$ by taking $\boldsymbol{S} = \mathit{paths}(\boldsymbol{G})$, $\mathfrak{c}'(\pi) = \mathfrak{c}(\mathit{tail}(\pi))$, $\mathfrak{a}'(\pi) = \mathfrak{a}(\mathit{tail}(\pi)) \cap \boldsymbol{qc}(C_0, \mathcal{R})$, for $\pi \in \mathit{paths}(\boldsymbol{G})$, and

$$\begin{aligned}\mathcal{E}(R) =& \{(\mathit{Root}(x), \mathit{Root}(y)) \mid y \text{ is an } R\text{-neighbour of } x\} \cup \\ & \{(u, \mathit{Root}(y)) \mid y \text{ is an } R\text{-neighbour of } \mathit{tail}(u)\} \cup \\ & \{(\mathit{Root}(x), u) \mid \mathit{tail}(u) \text{ is an } R\text{-neighbour of } x\} \cup \\ & \{(u, u) \mid \mathit{tail}(u) \text{ is an } R\text{-neighbour of } \mathit{tail}(u)\} \cup \\ & \{(u, v) \in \boldsymbol{S} \times \boldsymbol{S} \mid v = u, (\beta(y), y) \text{ and } y \text{ is an } R\text{-neighbour of } \mathit{tail}(u), \text{ or} \\ & \qquad u = v, (\beta(y), y) \text{ and } y \text{ is an } \mathit{inv}(R)\text{-neighbour of } \mathit{tail}(v)\}.\end{aligned}$$

We prove that $\boldsymbol{T}$ is a tableau for C_0 w.r.t. $\mathcal{R}$. Indeed, (p1) and (p14) follow from the initial step of the tableau algorithm and the fact that the labels of the root nodes are never removed; (p2) follows from the definition and the fact that the completion graph $\boldsymbol{G}$ is clash-free; (p3) follows from the rules creating new nodes and that $\boldsymbol{G}$ is clash-free; (p9) and (p16) follow from the definitions of $\mathcal{E}(R)$ and R-neighbour (and R-successor); (p6) and (p7) follow from the fact that the rules ($\sqcap$) and ($\sqcup$) are not applicable; (p12) follows from the definitions of $\mathcal{E}(R)$ and R-neighbour and the fact that the rule (guess) is not applicable; (p15) and (p5) follow from the definitions of $\mathcal{E}(R)$ and R-neighbour and that the completion graph $\boldsymbol{G}$ is clash-free; (p17) and (p18) follow from the fact that (r1) and (r3) are not applicable; (p19) from the fact that (r5^i) cannot be applied; and (p20) and (p23) follow from the definitions of $\mathcal{E}(R)$ and the fact that (r2) and (r6) are not applicable. The remaining cases are less straightforward. In some of them, we require the following:

($\sqcap$)	if $C_1 \sqcap C_2 \in \mathfrak{c}(x)$, x is not indirectly blocked and $\{C_1, C_2\} \not\subseteq \mathfrak{c}(x)$, then $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{C_1, C_2\}$
($\sqcup$)	if $C_1 \sqcup C_2 \in \mathfrak{c}(x)$, x is not indirectly blocked and $\{C_1, C_2\} \cap \mathfrak{c}(x) = \emptyset$, then $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{D\}$, for some $D \in \{C_1, C_2\}$
($\exists$)	if $\exists S.C \in \mathfrak{c}(x)$, x is not blocked and has no safe S-neighbour y with $C \in \mathfrak{c}(y)$ then create a new node $y \in V_2$ with $\mathfrak{l}(x, y) := \{S\}$, $\mathfrak{c}(y) := \{C, \top\}$, $\mathfrak{a}(y) := \emptyset$
(self)	if $\exists S.\mathit{Self} \in \mathfrak{c}(x)$, x is not blocked and x is not S-neighbour of x then add (x, x) to E_2, if it is not there yet, and set $\mathfrak{l}(x, x) := \mathfrak{l}(x, x) \cup \{S\}$
(guess)	if $(\leq nS.C) \in \mathfrak{c}(x)$, x is not indirectly blocked and there is an S-neighbour y of x such that $\{C, \neg C\} \cap \mathfrak{c}(y) = \emptyset$, then set $\mathfrak{c}(y) := \mathfrak{c}(y) \cup \{D\}$, for some $D \in \{C, \neg C\}$
($\geq$)	if $(\geq nS.C) \in \mathfrak{c}(x)$, x is not blocked and there are no distinct and safe $y_1, \ldots, y_n \in S^{\boldsymbol{G}}(x, C)$, then create n new successors $y_1, \ldots, y_n \in V_2$ of x; set $\mathfrak{l}(x, y_i) := \{S\}$, $\mathfrak{c}(y_i) := \{C, \top\}$, $\mathfrak{a}(y_i) := \emptyset$, $y_i \not\approx y_j$, for $1 \leq i < j \leq n$
($\leq$)	if $(\leq nS.C) \in \mathfrak{c}(x)$, x is not indirectly blocked, $\sharp S^{\boldsymbol{G}}(x, C) > n$ and there are $y, z \in S^{\boldsymbol{G}}(x, C)$ for which $y \not\approx z$ does not hold then (1) if z is a root node or an E_1-ancestor of y, then $\mathit{Merge}(y, z)$, (2) otherwise $\mathit{Merge}(z, y)$
(o)	if, for $o, o' \in \mathit{nom}(C_0)$, there is a node $y \neq x_o$ with $o' \in \mathfrak{c}(x_o) \cap \mathfrak{c}(y)$ and such that $x_o \not\approx y$ does not hold in the completion graph, then $\mathit{Merge}(y, x_o)$
($\leq_r$)	if $(\leq nS.C) \in \mathfrak{c}(x)$, $x \in V_1$, and there is an S-neighbour y of x such that $y \in V_2$, $(y, x) \in E_2$, $C \in \mathfrak{c}(y)$ and y is not indirectly blocked; and if there is no $n' \leq n$ with $(\leq n'S.C) \in \mathfrak{c}(x)$ and there are S-neighbours $z_1, \ldots, z_{n'} \in V_1$ of x with $C \in \mathfrak{c}(z_i)$ and $z_i \not\approx z_j$, for $1 \leq i, j \leq n'$, $i \neq j$, then (1) guess m, $1 \leq m \leq n$, and set $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{(\leq mS.C)\}$, (2) create m new nodes $y_1, \ldots, y_m \in V_1$ with $\mathfrak{l}(x, y_i) := \{S\}$, $\mathfrak{c}(y_i) := \{C, \top\}$, $\mathfrak{a}(y_i) := \emptyset$ and $y_i \not\approx y_j$, $1 \leq i < j \leq m$
($=_r$)	if $(\leq mS.C) \in \mathfrak{c}(x)$, $x \in V_1$, and there is an S-neighbour $y \in V_2$ of x with $C \in \mathfrak{c}(y)$, y is not indirectly blocked and there are S-neighbours $z_1, \ldots, z_m \in V_1$ of x with $C \in \mathfrak{c}(z_i)$ and $z_i \not\approx z_j$, for $1 \leq i, j \leq m$, $i \neq j$, and there is j_0, $1 \leq j_0 \leq m$ for which $y \not\approx z_{j_0}$ does not hold, then $\mathit{Merge}(y, z_{j_0})$

Table 1: Completion rules for the $\mathcal{SR}^{+}\mathcal{OIQ}$ tableau algorithm.

(r1)	if $\forall R.C \in \mathfrak{c}(x)$, x is not indirectly blocked and $\forall \mathfrak{A}^s_R.C \notin \mathfrak{a}(x)$, s the initial state, then $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall \mathfrak{A}^s_R.C\}$
(r2)	if $\forall \mathfrak{A}^p_R.C \in \mathfrak{a}(x)$, $q \in \delta_{\mathfrak{A}_R}(p, T)$ (where T can be ε), x is not indirectly blocked, y is a T-neighbour of x and $\forall \mathfrak{A}^q_R.C \notin \mathfrak{a}(y)$, then $\mathfrak{a}(y) := \mathfrak{a}(y) \cup \{\forall \mathfrak{A}^q_R.C\}$
(r3)	if $\forall \mathfrak{A}^a_R.C \in \mathfrak{a}(x)$, a an accepting state, x is not indirectly blocked and $C \notin \mathfrak{c}(x)$, then $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{C\}$
(r4^i)	if x is not indirectly blocked and there is $\mathfrak{C} \in \boldsymbol{qc}^*(\boldsymbol{r}_i)$ with $\{\mathfrak{C}, \neg\mathfrak{C}\} \cap \mathfrak{a}(x) = \emptyset$, then $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{D}\}$, for some $\mathfrak{D} \in \{\mathfrak{C}, \neg\mathfrak{C}\}$
(r5^i)	if x is not indirectly blocked, (r4^i) is not applicable, $\forall \mathfrak{A}^s_i.\mathfrak{C} \notin \mathfrak{a}(x)$, $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \mathfrak{a}(x))$, for the initial state s, then $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall \mathfrak{A}^s_i.\mathfrak{C}\}$
(r6)	if $\forall \mathfrak{A}^p.\mathfrak{C} \in \mathfrak{a}(x)$, x is not indirectly blocked, $q \in \delta_{\mathfrak{A}}(p, T)$ (where T can be ε), y is a T-neighbour of x and $\forall \mathfrak{A}^q.\mathfrak{C} \notin \mathfrak{a}(y)$, then $\mathfrak{a}(y) := \mathfrak{a}(y) \cup \{\forall \mathfrak{A}^q.\mathfrak{C}\}$
(r7)	if $\forall \mathfrak{A}^a.\mathfrak{C} \in \mathfrak{a}(x)$, a an accepting state, x is not indirectly blocked and $\mathfrak{C} \notin \mathfrak{a}(x)$, then $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{C}\}$
(r8)	if $\exists P.\mathfrak{C} \in \mathfrak{a}(x)$, x is not blocked and x has no safe P-neighbour y with $\mathfrak{C} \in \mathfrak{a}(y)$, then create a new node $y \in V_2$ and set $\mathfrak{l}(x, y) := \{P\}$, $\mathfrak{c}(y) := \{\top\}$, $\mathfrak{a}(y) := \{\mathfrak{C}\}$
(r9)	if $\mathfrak{C} \in \mathfrak{a}(x)$, $\mathfrak{C} = \bigvee_{j=1}^m \mathfrak{C}_j$, x is not indirectly blocked, $\{\mathfrak{C}_1, \dots, \mathfrak{C}_m\} \cap \mathfrak{a}(x) = \emptyset$, then $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{D}\}$, for some $\mathfrak{D} \in \{\mathfrak{C}_1, \dots, \mathfrak{C}_m\}$
(r10)	if $\mathfrak{C} \in \mathfrak{a}(x)$, for $\mathfrak{C} = (t^r, t^\forall, t^-)$, x is not indirectly blocked and $t^\forall \not\subseteq \mathfrak{a}(x)$, then set $\mathfrak{a}(x) := \mathfrak{a}(x) \cup t^\forall$

Table 2: Completion rules for the $\mathcal{SR}^+\mathcal{OIQ}$ tableau algorithm (cont.)

Proposition 24 *Suppose $u \in paths(\boldsymbol{G})$, $x = tail(u)$ and x has a safe R-neighbour $y \in V$. Then there is $v \in paths(\boldsymbol{G})$ with $(u, v) \in \mathcal{E}(R)$, $\mathfrak{c}'(v) = \mathfrak{c}(y)$ and $\mathfrak{a}'(v) = \mathfrak{a}(y) \cap \boldsymbol{qc}(C_0, \mathcal{R})$.*

Proof. As y is an R-neighbour of x, either $(x, y) \in E$ or $(y, x) \in E$. Four cases are possible:

- If $(x, y) \in E_1$, we set $v = u, (\beta(y), y)$.
- If $(x, y) \in E_2$, then $y \in V_1$ and we set $v = Root(y)$.
- If $(y, x) \in E_1$ then y is the only predecessor of x. In the case $tail'(u) = x$, there exists a path v such that $tail(v) = y$ and $u = v, (x, x)$; and in the case $tail'(u) \neq x$ (i.e., when x blocks $tail'(u)$), there exist a predecessor y' of $tail'(u)$ ($\mathfrak{c}(y') = \mathfrak{c}(y)$, $\mathfrak{a}(y') = \mathfrak{a}(y)$ and y' is an R-neighbour of $tail'(u)$) and a path v such that $tail(v) = y'$ and $u = v, (x, tail'(u))$.
- If $(y, x) \in E_2$ then $x \in V_1$, $u = Root(x)$. We set $v = Root(y)$ if $y \in V_1$. If $y \in V_2$ then y is not blocked (since y is safe), and so there exists a path v such that $tail(v) = y$.

In all of these cases, v is as required. ❑

(p4) If $\exists S.Self \in \mathfrak{c}'(u)$ then $\exists S.Self \in \mathfrak{c}(tail(u))$. Since (self) is not applicable, $tail(u)$ is an S-neighbour of $tail(u)$, and so $(u, u) \in \mathcal{E}(R)$.

(p8) If $\exists R.C \in \mathfrak{c}'(u)$ then $\exists R.C \in \mathfrak{c}(x)$, where $x = tail(u)$. Since ($\exists$) is not applicable, x has a safe R-neighbour y with $C \in \mathfrak{c}(y)$. By Proposition 24, there exists $v \in paths(\boldsymbol{G})$ such that $(u, v) \in \mathcal{E}(R)$ and $C \in \mathfrak{c}'(v)$.

(p10) If $\leq nS.C \in \mathfrak{c}'(u)$ then $\leq nS.C \in \mathfrak{c}(x)$, where $x = tail(u)$. Since the completion graph $\boldsymbol{G}$ is clash-free and ($\leq$) and ($=_r$) are not applicable, $\sharp S^{\boldsymbol{G}}(x, C) \leq n$. Suppose that $(u, v) \in \mathcal{E}(R)$ and $C \in \mathfrak{c}'(v)$. By the definition of $\mathcal{E}(R)$, the following cases are possible:

- $u = Root(x)$, $v = Root(y)$ and y is an R-neighbour of x. We have $y \in S^{\boldsymbol{G}}(x, C)$ and, since $y \in V_1$, there is no $v' \in paths(\boldsymbol{G})$ different from v and such that $y = tail(v')$ or $y = tail'(v')$.
- $x \in V_2$, $v = Root(y)$ and y is an R-neighbour of x. This case is considered analogously.
- $u = Root(x)$, $y = tail(v) \in V_2$, $v \neq u, (y, tail'(v))$ and y is an R-neighbour of x. This case is not possible since $\leq nS.C \in \mathfrak{c}(x)$, $x \in V_1$, $(y, x) \in E_2$ and the rules ($\leq_r$) and ($=_r$) are not applicable.
- $v = u$ and x is an R-neighbour of x. Then $x \in S^{\boldsymbol{G}}(x, C)$ and there is no $v' \in paths(\boldsymbol{G})$ different from u and such that $(u, v') \in \mathcal{E}(R)$ and $x = tail(v')$ or $x = tail'(v')$.
- $v = u, (\beta(y), y)$ and y is an R-neighbour of x. Then $y \in S^{\boldsymbol{G}}(x, C)$, $y \in V_2$ and x is the only predecessor of y. So there is no $v' \in paths(\boldsymbol{G})$ different from v and such that $(u, v') \in \mathcal{E}(R)$ and $y = tail(v')$ or $y = tail'(v')$.
- $u = v, (x, y)$, $x = \beta(y)$ and y is an $inv(R)$-neighbour of $tail(v)$. Then $tail(v) \in S^{\boldsymbol{G}}(x, C)$, $y \in V_2$ and $tail(v)$ is only one predecessor of y; so there is no $v' \in paths(\boldsymbol{G})$ different from v such that $(u, v') \in \mathcal{E}(R)$ and $tail(v) = tail(v')$ or $tail(v) = tail'(v')$.

Therefore, $\sharp\{v \in \boldsymbol{S} \mid (u,v) \in \mathcal{E}(S) \text{ and } C \in \mathfrak{c}(v)\} \leq \sharp S^{\boldsymbol{G}}(x, C) \leq n$.

(p11) If $(\geq nS.C) \in \mathfrak{c}'(u)$ then $(\geq nS.C) \in \mathfrak{c}(x)$, where $x = tail(u)$. Since $(\geq)$ is not applicable, x has safe S-neighbours $y_1, \dots, y_n$ with $C \in \mathfrak{c}(y_i)$ and $y_i \not\cong y_j$, for $1 \leq i, j \leq n$ and $j \neq i$. By Proposition 24, there exists $v_i \in paths(\boldsymbol{G})$ such that $(u, v_i) \in \mathcal{E}(S)$ and $C \in \mathfrak{c}'(v_i)$, for $1 \leq i \leq n$. In addition, there can be at most one y with $(y, x) \in E_1$ and, by the proof of Proposition 24, if $(y_i, x) \notin E_1$ then $tail(v_i) = y_i$ or $tail'(v_i) = y_i$. So, v_i and v_j are distinct for $i \neq j$, since $tail(v_i) \neq tail(v_j)$ or $tail'(v_i) \neq tail'(v_j)$ (in the case $(y_i, x) \in E_1$, $(x, y_j) \in E_1$ and y_i block y_j, i.e., $tail(v_i) = tail(v_j) = y_i$, we have $tail'(v_i) \neq tail'(v_j)$).

(p13) If $o \in \mathfrak{c}'(u) \cap \mathfrak{c}'(v)$ then $o \in \mathfrak{c}(tail(u))$, and so there is $o' \in nom(C_0)$ such that $x_{o'} = tail(u)$ and $u = Root(x_{o'})$. Similarly, there is $o'' \in nom(C_0)$ with $x_{o''} = tail(v)$ and $v = Root(x_{o''})$. Since the rule (o) is not applicable, $x_{o'} = x_{o''}$, and so $u = v$.

(p22) Let $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}'(u)$, where a is an accepting state and $\mathfrak{C} = \bigvee_{h=1}^{m_i} (t_h^r, t_h^\forall, t_h^-)$. Then $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(x)$, where $x = tail(u)$. Since (r7) cannot be applied, $\mathfrak{C} \in \mathfrak{a}(x)$, and since (r9) is not applicable, there is j such that $\mathfrak{C}_j = (t_j^r, t_j^\forall, t_j^-) \in \mathfrak{a}(x)$. Now, as (r10) is not applicable, we have $t_j^\forall \subseteq \mathfrak{a}(x)$; and since $\boldsymbol{G}$ is clash-free, $\mathfrak{a}(x)|^\forall_{inv(t_j^r)} \subseteq t_j^-$. Thus, $t_j^\forall \subseteq \mathfrak{a}'(u)$ and $\mathfrak{a}'(u)|^\forall_{inv(t_j^r)} \subseteq t_j^-$.

(p21) Let $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}'(u)$, where $\mathfrak{C} = \exists inv(P_{im_i}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-)$ and a is an accepting state. Then $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}(tail(u))$. We prove by induction on j that there is v_j such that $\exists inv(P_{ij}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-) \in \mathfrak{a}(tail(v_j))$. For $j = m_i$, set $v_{m_i} = u$. As (r7) is not applicable to $tail(v_{m_i})$, $\exists inv(P_{im_i}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-) \in \mathfrak{a}(tail(v_{m_i}))$, which establishes the induction basis. Assume now that our claim holds for j and prove it for $j-1$. As $tail(v_j)$ is not blocked and (r8) is not applicable, there is a safe $inv(P_{ij})$-neighbour y_{j-1} of $tail(v_j)$ such that $\exists inv(P_{i(j-1)}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-) \in \mathfrak{a}(y_{j-1})$. By Proposition 24, there is $v_{j-1} \in paths(\boldsymbol{G})$ with $(v_j, v_{j-1}) \in \mathcal{E}(inv(P_{ij}))$ and $\exists inv(P_{i(j-1)}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-) \in \mathfrak{a}(tail(v_{j-1}))$. For $j = 0$, we have $(t^r, t^\forall, t^-) \in \mathfrak{a}(tail(v_0))$. Further, as (r10) cannot be applied, we have $t^\forall \subseteq \mathfrak{a}(tail(v_0))$; and since $\boldsymbol{G}$ is clash-free, $\mathfrak{a}(tail(v_0))|^\forall_{inv(t^r)} \subseteq t^-$. Thus, $t^\forall \subseteq \mathfrak{a}'(v_0)$ and $\mathfrak{a}'(v_0)|^\forall_{inv(t^r)} \subseteq t^-$.

($\Rightarrow$) Take a tableau $\boldsymbol{T} = (\boldsymbol{S}, \mathfrak{c}', \mathfrak{a}', \mathcal{E})$ for C_0 w.r.t. $\mathcal{R}$ and extend it in the following way:

(e1) If $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}'(u)$, where $\mathfrak{C} = \exists inv(P_{im_i}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-)$ and a is an accepting state, then, by (p21), there are $v_0, v_1, \dots, v_{m_i} = u$ such that $(v_j, v_{j-1}) \in \mathcal{E}(inv(P_{ij}))$, for $1 \leq j \leq m_i$, $t^\forall \subseteq \mathfrak{a}'(v_0)$ and $\mathfrak{a}'(v_0)|^\forall_{inv(t^r)} \subseteq t^-$. In this case, we extend $\mathfrak{a}'(v_j)$ by taking $\mathfrak{a}'(v_j) := \mathfrak{a}'(v_j) \cup \{\exists inv(P_{ij}). \cdots \exists inv(P_{i1}).(t^r, t^\forall, t^-)\}$, for $0 \leq j \leq m_i$.

(e2) If $\forall \mathfrak{A}_i^a.\mathfrak{C} \in \mathfrak{a}'(u)$, where $\mathfrak{C} = \bigvee_{h=1}^{m_i} (t_h^r, t_h^\forall, t_h^-)$, then, by (p22), there is $j \in \{1, \dots, m_i\}$ such that $t_j^\forall \subseteq \mathfrak{a}'(u)$ and $\mathfrak{a}'(u)|^\forall_{inv(t_j^r)} \subseteq t_j^-$. In this case, we extend $\mathfrak{a}'(u)$ by taking $\mathfrak{a}'(u) := \mathfrak{a}'(u) \cup \{\mathfrak{C}, \mathfrak{C}_j\}$.

(e3) If there exists $\mathfrak{C} \in \bigcup_{i=0}^{l} \boldsymbol{qc}^*(\boldsymbol{r}_i)$ such that $\mathfrak{C} \notin \mathfrak{a}'(u)$, then $\mathfrak{a}'(u) := \mathfrak{a}'(u) \cup \{\neg\mathfrak{C}\}$.

(e4) If $(\leq nS.C) \in \mathfrak{c}'(u)$ and $S^{\boldsymbol{T}}(u, C) = \{v \in \boldsymbol{S} \mid (u,v) \in \mathcal{E}(S),\ C \in \mathfrak{c}(v)\} = \{v_1, \dots, v_m\}$ then, in view of (p10), we have $m \leq n$. In this case, we extend $\mathfrak{c}'(u)$ by taking $\mathfrak{c}'(u) := \mathfrak{c}'(u) \cup \{\leq mS.C\}$.

We now apply the completion rules using the extended tableau $\boldsymbol{T}$ so that in the end the algorithm obtains a clash-free completion graph $\boldsymbol{G} = (V_1, V_2, E_1, E_2, \mathfrak{c}, \mathfrak{a}, \mathfrak{l}, \not\cong)$ and returns 'yes'. For this purpose, we define a map $\mu\colon V \to \boldsymbol{S}$ and steer the applications of the non-deterministic completion rules in such a way that $\mathfrak{c}(x) \subseteq \mathfrak{c}'(\mu(x))$ and $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$, for all nodes $x \in V$ (cf. Horrocks, Kutz, & Sattler, 2005; Horrocks et al., 2006). Furthermore, we require that, for each pair of nodes x, y and each role R, if y is an R-successor of x, then $(\mu(x), \mu(y)) \in \mathcal{E}(R)$, and $x \not\approx y$ implies $\mu(x) \neq \mu(y)$. This will ensure that $\boldsymbol{G}$ is clash-free, since tableau $\boldsymbol{T}$ is clash-free.

We define μ by induction as follows. To begin with, by (p14), for each $o \in nom(C_0)$, there is some v_o with $o \in \mathfrak{c}'(v_o)$, and by (p1), there is some u_0 with $C_0 \in \mathfrak{c}'(u_0)$. The algorithm starts by constructing nodes x_o, for each $o \in nom(C_0)$, and x_{C_0} with $\mathfrak{c}(x_o) = \{o\}$ and $\mathfrak{c}(x_{C_0}) = \{C_0\}$. We set $\mu(x_o) = v_o$ and $\mu(x_{C_0}) = u_0$.

Observe that $\mathfrak{c}(x_o) \subseteq \mathfrak{c}'(\mu(x_o))$ and $\mathfrak{c}(x_{C_0}) \subseteq \mathfrak{c}'(\mu(x_{C_0}))$; also $\mathfrak{a}(x_o) = \emptyset \subseteq \mathfrak{a}'(\mu(x_o))$ and $\mathfrak{a}(x_{C_0}) = \emptyset \subseteq \mathfrak{a}'(\mu(x_{C_0}))$. We now consider applications of the completion rules.

If ($\sqcap$) can be applied to $x \in V$ with $C_1 \sqcap C_2 \in \mathfrak{c}(x)$, then $C_1 \sqcap C_2 \in \mathfrak{c}'(\mu(x))$, and so, by (p6), $C_1, C_2 \in \mathfrak{c}'(\mu(x))$. When we apply ($\sqcap$), C_1, C_2 are added to $\mathfrak{c}(x)$, so we again have $\mathfrak{c}(x) \subseteq \mathfrak{c}'(\mu(x))$.

If ($\sqcup$) can be applied to $x \in V$ with $C_1 \sqcup C_2 \in \mathfrak{c}(x)$, then $C_1 \sqcup C_2 \in \mathfrak{c}'(\mu(x))$, and so, by (p7), $\{C_1, C_2\} \cap \mathfrak{c}'(\mu(x)) \neq \emptyset$. We apply ($\sqcup$) so that $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{D\}$ for some $D \in \{C_1, C_2\} \cap \mathfrak{c}'(\mu(x))$, and again $\mathfrak{c}(x) \subseteq \mathfrak{c}'(\mu(x))$.

If ($\exists$) can be applied to $x \in V$ with $\exists S.C \in \mathfrak{c}(x)$, then $\exists S.C \in \mathfrak{c}'(\mu(x))$, and so, by (p8), there is a v with $(\mu(x), v) \in \mathcal{E}(S)$ and $C \in \mathfrak{c}'(v)$. By (p3), $\top \in \mathfrak{c}'(v)$ and, by (p16) for $S \sqsubseteq^* S'$, we have $(\mu(x), v) \in \mathcal{E}(S')$. We apply ($\exists$) so that a new node y is created with $\mathfrak{l}(x, y) := \{S\}$, $\mathfrak{c}(y) := \{C, \top\}$, $\mathfrak{a}(y) := \emptyset$ and $\mu(y) = v$. But then $\mathfrak{c}(y) \subseteq \mathfrak{c}'(\mu(y))$, $\mathfrak{a}(y) \subseteq \mathfrak{a}'(\mu(y))$ and $(\mu(x), \mu(y)) \in \mathcal{E}(S')$.

If (self) can be applied to $x \in V$ with $\exists S.\mathit{Self} \in \mathfrak{c}(x)$, then $\exists S.\mathit{Self} \in \mathfrak{c}'(\mu(x))$, and so, by (p4), $(\mu(x), \mu(x)) \in \mathcal{E}(S)$. By (p16) for $S \sqsubseteq^* S'$, we have $(\mu(x), \mu(x)) \in \mathcal{E}(S')$. We apply (self) by adding the arc (x, x), if it is not there yet, and setting $\mathfrak{l}(x, x) := \mathfrak{l}(x, x) \cup \{S\}$. Then we obtain $(\mu(x), \mu(x)) \in \mathcal{E}(S')$.

If (guess) can be applied to $x \in V$ with $(\leq nS.C) \in \mathfrak{c}(x)$ and an S-neighbour y of x, then $(\leq nS.C) \in \mathfrak{c}'(\mu(x))$, $(\mu(x), \mu(y)) \in \mathcal{E}(S)$, and so, by (p12), $\{C, \neg C\} \cap \mathfrak{c}'(\mu(y)) \neq \emptyset$. We apply (guess) so that $\mathfrak{c}(y) := \mathfrak{c}(y) \cup \{D\}$, for some $D \in \{C, \neg C\} \cap \mathfrak{c}'(\mu(y)$. Hence $\mathfrak{c}(y) \subseteq \mathfrak{c}'(\mu(y))$.

If ($\geq$) can be applied to $x \in V$ with $(\geq nS.C) \in \mathfrak{c}(x)$, then $(\geq nS.C) \in \mathfrak{c}'(\mu(x))$. By (p11), there are $v_1, \ldots, v_n \in S^{\boldsymbol{T}}(\mu(x), C)$, where $S^{\boldsymbol{T}}(u, C) = \{v \in \boldsymbol{S} \mid (u, v) \in \mathcal{E}(S),\ C \in \mathfrak{c}(v)\}$. We apply ($\geq$) by creating n new successors $y_1, \ldots, y_n$ of x and setting $\mathfrak{l}(x, y_i) := \{S\}$, $\mathfrak{c}(y_i) := \{C, \top\}$, $\mathfrak{a}(y_i) := \emptyset$, $y_i \not\approx y_j$ and $\mu(y_i) = v_i$, for $1 \leq i, j \leq n$, $j \neq i$. Then, for $S \sqsubseteq^* S'$, we have $(\mu(x), \mu(y_i)) \in \mathcal{E}(S')$ and also $\mathfrak{c}(y_i) \subseteq \mathfrak{c}'(\mu(y_i))$, for $1 \leq i \leq n$.

If ($\leq$) can be applied to $x \in V$ with $(\leq nS.C) \in \mathfrak{c}(x)$ and $\{y_1, \ldots, y_{n+1}\} \subseteq S^{\boldsymbol{G}}(x, C)$, then $(\leq nS.C) \in \mathfrak{c}'(\mu(x))$ and $\{\mu(y_1), \ldots, \mu(y_{n+1})\} \subseteq S^{\boldsymbol{T}}(\mu(x), C)$. By (p10), we have $\sharp S^{\boldsymbol{T}}(\mu(x), C) \leq n$, so there are j_1, j_2 such that $\mu(y_{j_1}) = \mu(y_{j_2}) = v$. Instead of y_{j_1}, y_{j_2}, we will write y, z; more precisely if y_{j_1} is a root or an E_1-ancestor of y_{j_2} then we set $z = y_{j_1}$ and $y = y_{j_2}$, otherwise we set $z = y_{j_2}$ and $y = y_{j_1}$. We apply $(\leq)$ by performing $Merge(y, z)$. Since $\mu(z) = v$, the required conditions on μ hold.

For $(=_r)$, the proof is similar to the previous case.

If (o) can be applied to $y \in V$ with $o' \in \mathfrak{c}(x_o) \cap \mathfrak{c}(y)$, for some $o, o' \in nom(C_0)$, then $o' \in \mathfrak{c}'(\mu(x_o)) \cap \mathfrak{c}'(\mu(y))$. By (p13), we have $\mu(x_o) = \mu(y)$, and therefore $\mathfrak{c}(x_o) \cup \mathfrak{c}(y) \subseteq \mathfrak{c}'(\mu(x_o)) \cup \mathfrak{c}'(\mu(y)) = \mathfrak{c}'(\mu(x_o))$. Similarly, $\mathfrak{a}(x_o) \cup \mathfrak{a}(y) \subseteq \mathfrak{a}'(\mu(x_o))$. We apply (o) by performing $Merge(y, x_o)$, so the required conditions for μ hold again.

If $(\leq_r)$ can be applied to $x \in V_1$ and an S-neighbour y of x with $(\leq nS.C) \in \mathfrak{c}(x)$, $y \in V_2$, $(y, x) \in E_2$ and $C \in \mathfrak{c}(y)$, then $(\leq nS.C) \in \mathfrak{c}'(\mu(x))$, $(\mu(x), \mu(y)) \in \mathcal{E}(S)$ and $C \in \mathfrak{c}'(\mu(y))$. By (p10), we have $\sharp S^{\boldsymbol{T}}(\mu(x), C) \leq n$, so $S^{\boldsymbol{T}}(\mu(x), C) = \{v_1, \ldots, v_m\}$, $m \leq n$. We apply $(\leq_r)$ so that $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{(\leq mS.C)\}$, create m new nodes $y_1, \ldots, y_m \in V_1$ with $\mathfrak{l}(x, y_i) := \{S\}$, $\mathfrak{c}(y_i) := \{C, \top\}$, $\mathfrak{a}(y_i) := \emptyset$, $y_i \not\cong y_j$ and $\mu(y_i) = v_i$ for all $1 \leq i \leq m$, $1 \leq j < i$. Then, for $S \sqsubseteq^* S'$, we have $(\mu(x), \mu(y_i)) \in \mathcal{E}(S')$, $\mathfrak{c}(y_i) \subseteq \mathfrak{c}'(\mu(y_i))$, for $1 \leq i \leq m$, and also, by (e4), $\mathfrak{c}(x) \subseteq \mathfrak{c}'(\mu(x))$.

If (r1) can be applied to $x \in V$ with $\forall R.C \in \mathfrak{c}(x)$, then $\forall R.C \in \mathfrak{c}'(\mu(x))$, and so, by (p17), $\forall \mathfrak{A}_R^s.C \in \mathfrak{a}'(\mu(x))$, where s is the initial state of $\mathfrak{A}_R$. We apply (r1) so that $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall \mathfrak{A}_R^s.C\}$. Clearly, we have $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$.

If (r2) can be applied to $x \in V$ with $\forall \mathfrak{A}_R^p.C \in \mathfrak{a}(x)$, $q \in \delta_{\mathfrak{A}_R}(p, T)$, and y is a T-neighbour of x, then $\forall \mathfrak{A}_R^p.C \in \mathfrak{a}'(\mu(x))$. If $T \neq \varepsilon$ then $(\mu(x), \mu(y)) \in \mathcal{E}(T)$ and, by (p20), $\forall \mathfrak{A}_R^q.C \in \mathfrak{a}'(\mu(y))$. If $T = \varepsilon$ then $y = x$ and, by (p23), $\forall \mathfrak{A}_R^q.C \in \mathfrak{a}'(\mu(y))$. In both cases we apply (r2) so that $\mathfrak{a}(y) := \mathfrak{a}(y) \cup \{\forall \mathfrak{A}_R^q.C\}$, and again $\mathfrak{a}(y) \subseteq \mathfrak{a}'(\mu(y))$.

If (r3) can be applied to $x \in V$ with $\forall \mathfrak{A}_R^a.C \in \mathfrak{a}(x)$, where a is an accepting state, then $\forall \mathfrak{A}_R^a.C \in \mathfrak{a}'(\mu(x))$. By (p18), $C \in \mathfrak{c}'(\mu(x))$. We apply (r3) so that $\mathfrak{c}(x) := \mathfrak{c}(x) \cup \{C\}$. Thus, $\mathfrak{c}(x) \subseteq \mathfrak{c}'(\mu(x))$.

If $(\mathrm{r}4^i)$ can be applied to $x \in V$ with $\mathfrak{C} \in \boldsymbol{qc}^*(\boldsymbol{r}_i)$, then, by (e3), $\{\mathfrak{C}, \neg\mathfrak{C}\} \cap \mathfrak{a}'(\mu(x)) \neq \emptyset$. We apply $(\mathrm{r}4^i)$ so that $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{D}\}$, for some $\mathfrak{D} \in \{\mathfrak{C}, \neg\mathfrak{C}\} \cap \mathfrak{a}'(\mu(x))$. Thus, $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$.

If $(\mathrm{r}5^i)$ can be applied to $x \in V$, then $\forall \mathfrak{A}_i^s.\mathfrak{C} \notin \mathfrak{a}(x)$, where s is the initial state of $\mathfrak{A}_i$ and $\mathfrak{C} = \Xi(\boldsymbol{r}_i, \mathfrak{a}(x))$. By (p19), $\forall \mathfrak{A}_i^s.\mathfrak{C}' \in \mathfrak{a}'(\mu(x))$, where $\mathfrak{C}' = \Xi(\boldsymbol{r}_i, \mathfrak{a}'(\mu(x)))$. Suppose $\mathfrak{C} \neq \mathfrak{C}'$. Since $\mathfrak{a}(x) \subseteq a'(\mu(x))$, there exists $\mathfrak{C}_1 \in \boldsymbol{qc}^*(\boldsymbol{r}_i)$ such that $\mathfrak{C}_1 \in \mathfrak{a}'(\mu(x))$ and $\mathfrak{C}_1 \notin \mathfrak{a}(x)$. As $(\mathrm{r}4^i)$ is not applicable, we have $\neg\mathfrak{C}_1 \in \mathfrak{a}(x)$, and so $\neg\mathfrak{C}_1 \in \mathfrak{a}'(\mu(x))$, which is a contradiction. Hence $\mathfrak{C} = \mathfrak{C}'$. We apply $(\mathrm{r}5^i)$ so that $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\forall \mathfrak{A}_i^s.\mathfrak{C}\}$. Thus, $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$.

If (r6) can be applied to $x \in V$ with $\forall \mathfrak{A}^p.\mathfrak{C} \in \mathfrak{a}(x)$, $q \in \delta_{\mathfrak{A}_R}(p, T)$, and y is a T-neighbour of x, then $\forall \mathfrak{A}^p.\mathfrak{C} \in \mathfrak{a}'(\mu(x))$. If $T \neq \varepsilon$ then $(\mu(x), \mu(y)) \in \mathcal{E}(T)$ and, by (p20), $\forall \mathfrak{A}^q.\mathfrak{C} \in \mathfrak{a}'(\mu(y))$. If $T = \varepsilon$ then $y = x$ and, by (p23), $\forall \mathfrak{A}^q.\mathfrak{C} \in \mathfrak{a}'(\mu(y))$. In either case, we apply (r6) so that $\mathfrak{a}(y) := \mathfrak{a}(y) \cup \{\forall \mathfrak{A}^q.\mathfrak{C}\}$. Thus, $\mathfrak{a}(y) \subseteq \mathfrak{a}'(\mu(y))$.

If (r7) can be applied to $x \in V$ with $\forall \mathfrak{A}^a.\mathfrak{C} \in \mathfrak{a}(x)$, where a is an accepting state, then $\forall \mathfrak{A}^a.\mathfrak{C} \in \mathfrak{a}'(\mu(x))$. By (e1) and (e2), $\mathfrak{C} \in \mathfrak{a}'(\mu(x))$. We apply (r7) in such a way that $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{C}\}$. Thus, $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$.

If (r8) can be applied to $x \in V$ with $\exists P.\mathfrak{C} \in \mathfrak{a}(x)$, then $\exists P.\mathfrak{C} \in \mathfrak{a}'(\mu(x))$, and so, by (e1), there is some v with $(\mu(x), v) \in \mathcal{E}(P)$ and $\mathfrak{C} \in \mathfrak{a}'(v)$. By (p16) for $P \sqsubseteq^* S'$, we have $(\mu(x), v) \in \mathcal{E}(S')$. We apply (r8) by creating a new node y with $\mathfrak{l}(x, y) := \{P\}$, $\mathfrak{c}(y) := \{\top\}$, $\mathfrak{a}(y) := \{\mathfrak{C}\}$ and $\mu(y) = v$. Thus, $\mathfrak{c}(y) \subseteq \mathfrak{c}'(\mu(y))$, $\mathfrak{a}(y) \subseteq \mathfrak{a}'(\mu(y))$ and $(\mu(x), \mu(y)) \in \mathcal{E}(S')$.

If (r9) can be applied to $x \in V$ with $\mathfrak{C} \in \mathfrak{a}(x)$, for $\mathfrak{C} = \bigvee_{j=1}^{m} \mathfrak{C}_j$, then $\mathfrak{C} \in \mathfrak{a}'(\mu(x))$. This means that $\mathfrak{C}$ is added to $\mathfrak{a}'(\mu(x))$ by (e2), and so, there is j such that $\mathfrak{C}_j \in \mathfrak{a}'(\mu(x))$. We apply (r9) so that $\mathfrak{a}(x) := \mathfrak{a}(x) \cup \{\mathfrak{C}_j\}$. Thus, $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$.

If (r10) can be applied to $x \in V$ with $\mathfrak{C} \in \mathfrak{a}(x)$, for $\mathfrak{C} = (t^r, t^\forall, t^-)$, then $\mathfrak{C} \in \mathfrak{a}'(\mu(x))$. This means that $\mathfrak{C}$ is added to $\mathfrak{a}'(\mu(x))$ by (e1) for $\mu(x) = v_0$, or by (e2). In either case, $t^\forall \subseteq \mathfrak{a}'(\mu(x))$. We apply (r10) so that $\mathfrak{a}(x) := \mathfrak{a}(x) \cup t^\forall$. Thus, $\mathfrak{a}(x) \subseteq \mathfrak{a}'(\mu(x))$.

This completes the proof of the lemma. ❑

As an immediate consequence of Lemmas 19, 22 and 23, we obtain our main Theorem 15 according to which concept satisfiability w.r.t. $\mathcal{SR}^+\mathcal{OIQ}$ KBs is decidable. It is worth noting that if the given RBox $\mathcal{R}$ does not contain RAs of the form (C)–(F) then our tableau algorithm behaves in the same way as the algorithm for $\mathcal{SROIQ}$ (Horrocks et al., 2006). However, if $\mathcal{R}$ contains one RA of the form (C)–(F) the algorithm will have to construct the set $\boldsymbol{qc}(C_0, \mathcal{R})$ of quasi-concepts, which contains subsets of the previously constructed sets of quasi-concepts $\boldsymbol{qc}(\boldsymbol{r}_0)$, and so may suffer an exponential blow-up. More precisely, the new quasi-concepts in $\boldsymbol{qc}(C_0, \mathcal{R})$ are built from triples of the form $(t^r, t^\forall, t^-)$, where $t^\forall \subseteq \boldsymbol{qc}(\boldsymbol{r}_0)|^\forall_{t^r}$ and $t^- \subseteq \boldsymbol{qc}(\boldsymbol{r}_0)|^\forall_{inv(t^r)}$. Furthermore, the algorithm may suffer one more exponential blow-up every time we add an extra RA of the form (C)–(F) and extend the sequence $\boldsymbol{r}_{i_1} \lhd \boldsymbol{r}_{i_2}, \boldsymbol{r}_{i_2} \lhd \boldsymbol{r}_{i_3}, \ldots, \boldsymbol{r}_{i_{h-1}} \lhd \boldsymbol{r}_{i_h}$ because again the set of quasi-concepts may become exponentially larger.

References

Baader, F. (2003). Restricted role-value-maps in a description logic with existential restrictions and terminological cycles. In *Proccedings of the 2003 International Workshop on Description Logics (DL2003).*

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

Baldoni, M. (1998). *Normal Multimodal Logics: Automatic Deduction and Logic Programming Extension.* Ph.D. thesis, Universit'a degli Studi di Torino.

Bock, C., Zha, X., Suh, H., & Lee, J. (2010). Ontological product modeling for collaborative design. *Advanced Engineering Informatics*, *24*, 510–524.

Bock, C. (2004). UML 2 Composition Model. *Journal of Object Technology*, *3*(10), 47–74.

Brachman, R. J., & Schmolze, J. G. (1985). An overview of the KL-ONE knowledge representation system. *Cognitive Science*, *9*(2), 171–216.

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

del Cerro, L. F., & Penttonen, M. (1988). Grammar logics. *Logique et Analyse*, *121*, 123–134.

Demri, S. (2001). The complexity of regularity in grammar logics and related modal logics. *Journal of Logic and Computation*, *11*(6), 933–960.

Halpern, J., & Moses, Y. (1992). A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, *54*(2), 319–379.

Horrocks, I., Kutz, O., & Sattler, U. (2005). The irresistible $\mathcal{SRIQ}$. In *Proc. of the First OWL Experiences and Directions Workshop*.

Horrocks, I., Kutz, O., & Sattler, U. (2006). The even more irresistible $\mathcal{SROIQ}$. In *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2006)*, pp. 57–67.

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

Horrocks, I., & Sattler, U. (2007). A tableau decision procedure for $\mathcal{SHOIQ}$. *Journal of Automated Reasoning*, *39*(3), 249–276.

Kazakov, Y. (2010). An extension of complex role inclusion axioms in the description logic $\mathcal{SROIQ}$. In *Proceedings of IJCAR*, pp. 472–486.

Krdzavac, N., & Bock, C. (2008). Reasoning in manufacturing part-part examples with OWL2. Tech. rep., U.S. National Institute of Standards and Technology, Gaithersburg, United States of America.

Motik, B., Cuenca Grau, B., Horrocks, I., & Sattler, U. (2009). Representing ontologies using description logics, description graphs, and rules. *Artificial Intelligence*, *173*(14), 1275–1309.

Motik, B., Shearer, R., & Horrocks, I. (2009). Hypertableau reasoning for description logics. *Journal of Artificial Intelligence Research (JAIR)*, *36*(1), 165–228.

Rector, A. (2002). Analysis of propagation along transitive roles: formalisation of the GALEN experience with medical ontologies. In *Proceedings of the International Workshop on Description Logics 2002 (DL2002)*.

Schmidt-Schauß, M. (1989). Subsumption in KL-ONE is undecidable. In *Proccedings of the 1st Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'89)*, pp. 421–431.

Simančík, F. (2012). Elimination of complex RIAs without automata. In *Proceedings of the International Workshop on Description Logics 2012 (DL2012)*.

Tseitin, G. (1956). Associative calculi with undecidable equivalence problems. *Dokl. Akad. Nauk SSSR*, *107*(3), 370–371. (In Russian).

Wessel, M. (2001). Obstacles on the way to qualitative spatial reasoning with description logics: some undecidability results. In *Proceedings of the International Workshop on Description Logics (DL2001)*.

Wessel, M. (2002). On spatial reasoning with description logics. In *Proceedings of the International Workshop on Description Logics 2002 (DL2002)*.