$\mathcal{ALC}_{\mathcal{RA}}$ – $\mathcal{ALC}$ with Role Axioms

Michael Wessel, Volker Haarslev, and Ralf Möller
University of Hamburg, Computer Science Department,
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
Email: {mwessel,haarslev,moeller}.informatik.uni-hamburg.de

Abstract

This paper presents a tableaux calculus for deciding the concept satisfiability problem of the new description logic $\mathcal{ALC}_{\mathcal{RA}}$ and discusses some open problems. $\mathcal{ALC}_{\mathcal{RA}}$ augments the description logic $\mathcal{ALC}$ with role inclusion axioms of the form $S \circ T \sqsubseteq R_1 \sqcup \ldots \sqcup R_n$. Additionally, all roles are interpreted as disjoint.

1 Introduction and Motivation

In the following we present the new description logic $\mathcal{ALC}_{\mathcal{RA}}$ and a tableaux calculus for deciding the $\mathcal{ALC}_{\mathcal{RA}}$ concept satisfiability problem. The paper presents work in progress. $\mathcal{ALC}_{\mathcal{RA}}$ augments the description logic $\mathcal{ALC}$ with role inclusion axioms of the form $S \circ T \sqsubseteq R_1 \sqcup \ldots \sqcup R_n$, $n \geq 1$, enforcing $S^{\mathcal{I}} \circ T^{\mathcal{I}} \subseteq R_1^{\mathcal{I}} \cup \ldots \cup R_n^{\mathcal{I}}$ on the interpretations $\mathcal{I}$. A finite set of these role axioms is called a role box, $\mathfrak{R}$. Additionally, *all* roles have to be interpreted as disjoint. As a first example, consider the $\mathcal{ALC}_{\mathcal{RA}}$ concept $(\exists R.\exists S.C) \sqcap \forall T.\neg C$ w.r.t. the role axiom $R \circ S \sqsubseteq T$. This concept is unsatisfiable in $\mathcal{ALC}_{\mathcal{RA}}$, but satisfiable in $\mathcal{ALC}$. $\mathcal{ALC}_{\mathcal{RA}}$ is at least as expressive as $\mathcal{ALC}_{\mathcal{R}^+}$, since a role R can be declared as *transitively closed* with the role axiom $R \circ R \sqsubseteq R$. As another example taken from the realm of genealogy, let us consider the concept term $(\exists brother.\exists sister.\exists sister.\exists daughter.\exists sister.css) \sqcap \forall niece.\neg css$ w.r.t. the role box $\{brother \circ sister \sqsubseteq sister, sister \circ daughter \sqsubseteq niece, daughter \circ sister \sqsubseteq daughter, sister \circ sister \sqsubseteq sister\}$. A careful inspection will reveal that this concept is inconsistent since the computer science student (css) plays also the role of a niece and is therefore a filler of the *niece* role. Note that composition of roles is not allowed to appear on the right hand side of role axioms. One can therefore not write axioms like $niece \sqsubseteq (brother \circ daughter) \sqcup (sister \circ daughter)$. It is easy to show that allowing composition on the right hand side of the role axioms yields a form of undecidable role-value maps (see [2]).

We believe that in many application domains disjoint roles are an indispensable tool required for an adequate modeling. Reconsidering our genealogical example, no individual can play the role of a nephew *and* a niece. If the role box given above would additionally contain the axiom $\mathit{niece} \circ \mathit{sister} \sqsubseteq \mathit{nephew}$, the concept $\exists \mathit{sister}.\exists \mathit{daughter}.\exists \mathit{sister}.\top$ would become inconsistent w.r.t. this role box.

2 Syntax and Semantics

The set of concept terms (concepts for short) is the same as for the language $\mathcal{ALC}$. Let $\mathcal{N_C}$ be the set of concept names, and $\mathcal{N_R}$ be the set of role names (roles for short), with $\mathcal{N_C} \cap \mathcal{N_R} = \emptyset$. Now, every $C \in \mathcal{N_C}$ is a concept term. Additionally, if C, C_1, C_2 are concept terms, and $R \in \mathcal{N_R} \setminus \{R_?\}$, then also $C_1 \sqcap C_2$, $C_1 \sqcup C_2$, $\exists R.C$, $\forall R.C$ are concept terms. Note that the special role $R_? \in \mathcal{N_R}$ cannot be used within the concept terms. $R_?$ is the so-called *don't care role*. Its purpose will be explained later.[1] $\bot$ ($\top$) is an abbreviation for $C \sqcap \neg C$ ($C \sqcup \neg C$), for some C. The function $\mathsf{roles}(C)$ returns the set of role names used in C (e.g. $\mathsf{roles}(\exists R.\exists S.C \sqcap \exists T.D) = \{R, S, T\}$), and $\mathsf{sub}(C)$ returns the subconcepts of C (e.g. $\mathsf{sub}(C \sqcap \forall R.D) = \{C \sqcap \forall R.D, C, \forall R.D, D\}$). The syntax of role axioms is as follows:

Definition 1 (Role Axioms, Role Box) If $S, T, R_1, \ldots, R_n \in \mathcal{N_R}$, then the expression $S \circ T \sqsubseteq R_1 \sqcup \ldots \sqcup R_n$, $n \geq 1$, is called a *role axiom*. If $ra = S \circ T \sqsubseteq R_1 \sqcup \ldots \sqcup R_n$, then $\mathsf{pre}(ra) =_{def} (S, T)$ and $\mathsf{con}(ra) =_{def} \{R_1, \ldots, R_n\}$. A finite set $\mathfrak{R}$ of role axioms is called a *role box*.

Let $\mathsf{roles}(ra) =_{def} \{S, T, R_1, \ldots, R_n\}$, and $\mathsf{roles}(\mathfrak{R}) =_{def} \bigcup_{ra \in \mathfrak{R}} \mathsf{roles}(ra)$. If C is an $\mathcal{ALC_{RA}}$ concept and $\mathfrak{R}$ is a role box, we also use the function $\mathsf{roles}(C, \mathfrak{R}) =_{def} \mathsf{roles}(C) \cup \mathsf{roles}(\mathfrak{R})$.

The role box $\mathfrak{R}$ is said to be *admissible* iff $R_? \notin \mathsf{roles}(\mathfrak{R})$, and $\forall ra_1, ra_2 \in \mathfrak{R}$: $\mathsf{pre}(ra_1) = \mathsf{pre}(ra_2) \Rightarrow ra_1 = ra_2$. We can then use the function $\mathsf{ra}(S, T) = ra$ to refer to this unique role axiom (if $\mathfrak{R}$ is clear from the context) and define $\mathsf{con}(S, T) =_{def} \mathsf{con}(\mathsf{ra}(S, T))$. In the following, we will only consider admissible role boxes. Additionally, the *completion* w.r.t. the concept term C of the role box $\mathfrak{R}$ is the role box $\mathfrak{R}(C) =_{def} \mathfrak{R} \cup \{\, R \circ S \sqsubseteq \sqcup_{T \in (\{R_?\} \cup \mathsf{roles}(C, \mathfrak{R}))} T \mid$
$\neg(\exists ra \in \mathfrak{R} : \mathsf{pre}(ra) = (R, S)), R, S \in (\{R_?\} \cup \mathsf{roles}(C, \mathfrak{R}))\,\}$.
This role box is also called the completed role box of $\mathfrak{R}$ w.r.t. C.

Definition 2 (Interpretation) An interpretation $\mathcal{I} =_{def} (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set $\Delta^{\mathcal{I}}$, called the domain of $\mathcal{I}$, and an interpretation function $\cdot^{\mathcal{I}}$ that maps every concept name to a subset of $\Delta^{\mathcal{I}}$, and every role name to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Additionally, for all roles $R, S \in \mathcal{N_R}$, $R \neq S$: $R^{\mathcal{I}} \cap S^{\mathcal{I}} = \emptyset$. The following functions on $\mathcal{I}$ will be used: The *universal relation* of $\mathcal{I}$ is defined

[1] More specifically, $R_?$ is needed for the definition of the *completed role box*. A complete role box is needed for the presented tableaux algorithm in order to be sound and complete.

as $\mathcal{UR}(\mathcal{I}) =_{def} \bigcup_{R \in \mathcal{N_R}} R^{\mathcal{I}}$, and the universal relation w.r.t. a set of role names $\mathcal{R}$ as $\mathcal{UR}(\mathcal{I},\mathcal{R}) =_{def} \bigcup_{R \in \mathcal{R}} R^{\mathcal{I}}$. The *skeleton* of $\mathcal{I}$ is defined as $\mathcal{SKEL}(\mathcal{I}) =_{def} \mathcal{UR}(\mathcal{I}) \setminus (\mathcal{UR}(\mathcal{I})^+ \circ \mathcal{UR}(\mathcal{I})^+)$, and the skeleton w.r.t. a set of role names $\mathcal{R}$ as $\mathcal{SKEL}(\mathcal{I},\mathcal{R}) =_{def} \mathcal{UR}(\mathcal{I},\mathcal{R}) \setminus (\mathcal{UR}(\mathcal{I},\mathcal{R})^+ \circ \mathcal{UR}(\mathcal{I},\mathcal{R})^+)$. If $<i,j> \in \mathcal{SKEL}(\mathcal{I})$, the edge is called a *direct edge*, otherwise an *indirect edge*. If $<i,j> \in \mathcal{UR}(\mathcal{I})$, the edge is called an *incoming edge* for j. The interpretation function $\cdot^{\mathcal{I}}$ can be extended to arbitrary concepts C: $(\neg C)^{\mathcal{I}} =_{def} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} =_{def} C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $(C \sqcup D)^{\mathcal{I}} =_{def} C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $(\exists R.C)^{\mathcal{I}} =_{def} \{ i \in \Delta^{\mathcal{I}} \mid \exists j \in C^{\mathcal{I}} : <i,j> \in R^{\mathcal{I}} \}$, $(\forall R.C)^{\mathcal{I}} =_{def} \{ i \in \Delta^{\mathcal{I}} \mid \forall j : <i,j> \in R^{\mathcal{I}} \Rightarrow j \in C^{\mathcal{I}} \}$. An interpretation $\mathcal{I}$ satisfies resp. is a model of a concept term C, $\mathcal{I} \models C$, iff $C^{\mathcal{I}} \neq \emptyset$. An interpretation $\mathcal{I}$ satisfies resp. is a model of a role axiom $S \circ T \sqsubseteq R_1 \sqcup \ldots \sqcup R_n$, $\mathcal{I} \models S \circ T \sqsubseteq R_1 \sqcup \ldots \sqcup R_n$, iff $S^{\mathcal{I}} \circ T^{\mathcal{I}} \subseteq R_1^{\mathcal{I}} \cup \ldots \cup R_n^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a model of a role box $\mathfrak{R}$, $\mathcal{I} \models \mathfrak{R}$, iff for all role axioms $ra \in \mathfrak{R}$: $\mathcal{I} \models ra$. An interpretation $\mathcal{I}$ is a model of $(C, \mathfrak{R})$, $\mathcal{I} \models (C, \mathfrak{R})$, iff it is a model of C and a model of $\mathfrak{R}$: $\mathcal{I} \models C$, $\mathcal{I} \models \mathfrak{R}$. Let us collect some facts:

Proposition 1 If $\mathcal{I} \models (C, \mathfrak{R})$, then for every interpretation $\mathcal{I}' = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}'})$ with $D^{\mathcal{I}'} = D^{\mathcal{I}}$ for all concept names $D \in \mathsf{sub}(C) \cap \mathcal{N_C}$ and $R^{\mathcal{I}'} = R^{\mathcal{I}}$ for all role names $R \in \mathsf{roles}(C, \mathfrak{R})$: $\mathcal{I}' \models (C, \mathfrak{R})$.

Proposition 2 $(C, \mathfrak{R})$ is satisfiable iff $(\mathsf{NNF}(C), \mathfrak{R})$ is satisfiable.[2]

Proposition 3 $(C, \mathfrak{R})$ is satisfiable iff $(C, \mathfrak{R}(C))$ is satisfiable.

In the proof (see [3]), a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $(C, \mathfrak{R})$ is augmented to become a model $\mathcal{I}' = (\Delta^{\mathcal{I}'}, \cdot^{\mathcal{I}'})$ of $(C, \mathfrak{R}(C))$ (the other direction is obvious). Let $\Delta^{\mathcal{I}'} =_{def} \Delta^{\mathcal{I}}$, and $\cdot^{\mathcal{I}'}(D) =_{def} D^{\mathcal{I}}$ for all concept names $D \in \mathcal{N_C}$, and $\cdot^{\mathcal{I}'}(R) =_{def} R^{\mathcal{I}}$ for all role names $R \in \mathsf{roles}(C, \mathfrak{R})$, and $\cdot^{\mathcal{I}'}(R) =_{def} \emptyset$ for all role names $R \in ((\mathcal{N_R} \setminus \{R_?\}) \setminus \mathsf{roles}(C, \mathfrak{R}))$, and $\cdot^{\mathcal{I}'}(R_?) =_{def} \mathcal{UR}(\mathcal{I}, \mathsf{roles}(C, \mathfrak{R}))^+ \setminus \mathcal{UR}(\mathcal{I}, \mathsf{roles}(C, \mathfrak{R}))$. In other words: the don't care relationship is established between two domain objects if there is no edge connecting them. It is then shown that all "old" role axioms are still satisfied (those in $\mathfrak{R}$), and all "new" role axioms (those in $\mathfrak{R}(C) \setminus \mathfrak{R}$) are also satisfied. Also note that $\mathcal{I}'$ is a *connected* model. In fact, every individual is connected to all its ancestors via exactly one role.

Proposition 4 $\mathcal{ALC}_{\mathcal{RA}}$ does not have the *finite model property*, i.e. there are pairs $(C, \mathfrak{R})$ that have no finite models.

Proof 1 Consider $(\exists R.\exists R.\top) \sqcap (\forall S.\exists R.\top)$ w.r.t. $\{R \circ R \sqsubseteq S, R \circ S \sqsubseteq S, S \circ R \sqsubseteq S, S \circ S \sqsubseteq S\}$. Assume there is a finite model $\mathcal{I}$ with $n = |\Delta^{\mathcal{I}}|$. Let

[2] $\mathsf{NNF}(C)$ returns the *Negation Normal Form* of C. The NNF can be obtained by "pushing the negation sign inwards", e.g. by exhaustively applying the rules $\neg(C_1 \sqcap C_2) \rightarrow \neg C_1 \sqcup \neg C_2$, $\neg(C_1 \sqcup C_2) \rightarrow \neg C_1 \sqcap \neg C_2$, $\neg \forall R.C_1 \rightarrow \exists R.\neg C_1$ and $\neg \exists R.C_1 \rightarrow \forall R.\neg C_1$.

$i_0 \in ((\exists R.\exists R.\top) \sqcap (\forall S.\exists R.\top))^{\mathcal{I}}$. Due to $i_0 \in (\exists R.\exists R.\top)^{\mathcal{I}}$ there must be some i_1, i_2 with $<i_0, i_1> \in R^{\mathcal{I}}$, $<i_1, i_2> \in R^{\mathcal{I}}$, $<i_0, i_2> \in S^{\mathcal{I}}$ due to $R \circ R \sqsubseteq S$. Since $i_0 \in (\forall S.\exists R.\top)^{\mathcal{I}}$ we have $i_2 \in (\exists R.\top)^{\mathcal{I}}$, and there must be some i_3 with $<i_2, i_3> \in R^{\mathcal{I}}$, $<i_0, i_3> \in S^{\mathcal{I}}$ due to $S \circ R \sqsubseteq S$, and $<i_1, i_3> \in S^{\mathcal{I}}$ due to $R \circ R \sqsubseteq S$. Since $i_0 \in (\forall S.\exists R.\top)^{\mathcal{I}}$ we have $i_3 \in (\exists R.\top)^{\mathcal{I}}$, etc., until we reach i_{n-1} with $<i_0, i_{n-1}> \in S^{\mathcal{I}}$, $<i_{n-2}, i_{n-1}> \in R^{\mathcal{I}}$, $i_{n-1} \in (\exists R.\top)^{\mathcal{I}}$. Due to $n = |\Delta^{\mathcal{I}}|$, we have to "reuse" one of the individuals in $\Delta^{\mathcal{I}}$ when "creating" the R-successor of i_{n-1}: $<i_{n-1}, i_j> \in R^{\mathcal{I}}$ for some $j \in 0 \ldots n-1$. If $j = n-1$, then $<i_{n-1}, i_{n-1}> \in R^{\mathcal{I}}$, and due to $R \circ R \sqsubseteq S$: $<i_{n-1}, i_{n-1}> \in S^{\mathcal{I}}$. If $j = n-2$, then $<i_{n-1}, i_{n-2}> \in R^{\mathcal{I}}$, $<i_{n-2}, i_{n-1}> \in R^{\mathcal{I}}$, $<i_{n-1}, i_{n-1}> \in S^{\mathcal{I}}$ ($R \circ R \sqsubseteq S$), and finally $<i_{n-2}, i_{n-1}> \in S^{\mathcal{I}}$ ($R \circ S \sqsubseteq S$). If $j < n-2$, then $<i_{n-1}, i_j> \in R^{\mathcal{I}}$, $<i_j, i_{n-1}> \in S^{\mathcal{I}}$, $<i_j, i_j> \in S^{\mathcal{I}}$ ($S \circ R \sqsubseteq S$): now, either $j \neq 0$, then $<i_{j-1}, i_j> \in R^{\mathcal{I}}$, $<i_j, i_j> \in S^{\mathcal{I}}$, $<i_{j-1}, i_j> \in S^{\mathcal{I}}$ ($R \circ S \sqsubseteq S$), or $j = 0$, $<i_0, i_0> \in S^{\mathcal{I}}$, $<i_0, i_1> \in R^{\mathcal{I}}$, $<i_0, i_1> \in S^{\mathcal{I}}$ ($S \circ R \sqsubseteq S$). In all cases, we have $R^{\mathcal{I}} \cap S^{\mathcal{I}} \neq \emptyset$. Since the argumentation applies independently of n, there can be no finite models. □

3 A Tableaux Algorithm

In a similar way as for other description logics, a non-deterministic tableaux algorithm is given that constructs a so-called finite *completion tree*. Soundness of the algorithm is proven by showing that a so-called *tableau* can be constructed from a complete and clash-free completion tree that has been generated by the algorithm. Completeness is proven by showing how to construct a clash-free completion tree from a given tableau. Basically, a tableau is a possibly infinite tree whose edges are labeled with role names, and whose nodes are labeled with constraints enforced on these nodes (see [1, 3]). A tableau for $(\mathsf{NNF}(C), \mathfrak{R}(C))$ is just an other representation of a special model $\mathcal{I}$ of $(\mathsf{NNF}(C), \mathfrak{R}(C))$. We call these models "tree skeleton models". Let $\mathcal{I}$ be the (tree skeleton) model corresponding to some given tableau. Then, $\mathcal{SKEL}(\mathcal{I})$ corresponds to the labeled tree of this tableau, e.g. if a node y in the tableau is an R-successor of the node x, $<x, y> \in \mathcal{E}_{\mathcal{R}}$, due to some constraint $\exists R \ldots$ enforced on node x, $\exists R \ldots \in \mathcal{L}_{\mathcal{N}}(x)$, then w.r.t. $\mathcal{I}$ we have $<x, y> \in R^{\mathcal{I}} \cap \mathcal{SKEL}(\mathcal{I})$. However, the *indirect edges* which might be present in $\mathcal{I}$ due to role axioms cannot be represented in the tableau in this way, since a tableau is a labeled *tree*. For example, if $R \circ S \sqsubseteq T$, then a model $\mathcal{I}$ with $<x_0, x_1> \in R^{\mathcal{I}}$, $<x_1, x_2> \in S^{\mathcal{I}}$ must satisfy $<x_0, x_2> \in T^{\mathcal{I}}$. Therefore, every incoming edge for a node x in the model is represented in the tableau by a special *annotated all constraint* of the form $(\forall U.D)_{S,w} \in \mathcal{L}_{\mathcal{N}}(x)$, where S represents the type of the incoming edge, and w is a word of role names denoting a path in the tree leading from the individual from which the edge originates to x. In our example we would have $(\forall \ldots)_{T,RS} \in \mathcal{L}_{\mathcal{N}}(x_2)$ due to $<x_0, x_2> \in T^{\mathcal{I}}$, $<x_0, x_1> \in \mathcal{E}_{\mathcal{R}}$, $<x_1, x_2> \in \mathcal{E}_{\mathcal{S}}$, and $(\forall \ldots)_{S,S} \in \mathcal{L}_{\mathcal{N}}(x_2)$ due to $<x_1, x_2> \in S^{\mathcal{I}}$, $<x_1, x_2> \in \mathcal{E}_{\mathcal{S}}$. Assume that $\forall U.D \in \mathcal{L}_{\mathcal{N}}(x_0)$. Then, the presence of the constraint $(\forall U.D)_{T,RS} \in \mathcal{L}_{\mathcal{N}}(x_2)$ is ensured (see below). Since x_2 is an indirect T-successor of x_0 and not a U-successor, $D \notin \mathcal{L}_{\mathcal{N}}(x_2)$. If we additionally had $\forall T.D \in \mathcal{L}_{\mathcal{N}}(x_0)$, then also $(\forall T.D)_{T,RS} \in \mathcal{L}_{\mathcal{N}}(x_2)$, and $D \in \mathcal{L}_{\mathcal{N}}(x_2)$. Whenever a constraint $(\forall U.D)_{T,w} \in \mathcal{L}_{\mathcal{N}}(x)$ with $U = T$ is encountered, $D \in \mathcal{L}_{\mathcal{N}}(x)$ is ensured.

Definition 3 (Tableau) If C is an $\mathcal{ALC}_{\mathcal{RA}}$ concept in NNF and $\mathfrak{R}$ is a role box, a *tableau* $\mathcal{T}$ for $(C, \mathfrak{R}(C))$ is a tuple $(\mathcal{N}, \mathcal{L}_{\mathcal{N}}, \mathcal{E}, \mathcal{L}_{\mathcal{E}})$, where $\mathcal{N}$ is a set of nodes, $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is a set of edges, and the total labeling function $\mathcal{L}_{\mathcal{E}} : \mathcal{E} \rightarrow \mathsf{roles}(\mathfrak{R})$ associates edges with role names. For a role $R \in \mathcal{N}_{\mathcal{R}}$, the set of R-edges is $\mathcal{E}_{\mathcal{R}} =_{def} \{ \langle x, y \rangle \mid (\langle x, y \rangle, R) \in \mathcal{L}_{\mathcal{E}} \}$. The graph $(\mathcal{N}, \mathcal{E})$ has the structure of a possibly infinite tree. Additionally, $\mathcal{L}_{\mathcal{N}}$ is a node labeling function: $\mathcal{L}_{\mathcal{N}} : \mathcal{N} \rightarrow \mathsf{sub}(C) \cup \mathcal{Q}$, where $\mathcal{Q} = \{ (\forall R.C_1)_{S,w} \mid \forall R.C_1 \in \mathsf{sub}(C),$ or $\forall R.C_1 = \forall R_?.\top,$
$R, S \in \mathsf{roles}(\mathfrak{R}(C)), w \in \mathsf{roles}(\mathfrak{R})^+ \}$.
If $\langle x, y \rangle \in \mathcal{E}$, y is called a *successor* of x. If $\langle x, y \rangle \in \mathcal{E}^+$, x is called an *ancestor* of y, and y is called a *descendant* of x. Let $\mathsf{w\_ancestor}(y, w) =_{def} x$ iff $w = R_1 R_2 \ldots R_n$, where $w \in \mathsf{roles}(\mathfrak{R})^+$, $\langle x, x_1 \rangle \in \mathcal{E}_{R1}$, $\langle x_1, x_2 \rangle \in \mathcal{E}_{R2}$, $\ldots$, $\langle x_{n-1}, y \rangle \in \mathcal{E}_{Rn}$. Additionally, the following conditions hold:

1. There is some node $x_0 \in \mathcal{N}$ with $C \in \mathcal{L}_{\mathcal{N}}(x_0)$.
2. For all $x, y \in \mathcal{N}$, and for all $C_1, C_2, C_3 \in \mathsf{sub}(C)$ and for all $(\forall R_i.C_i)_{S_i,w} \in \mathcal{Q}$ and $R, R_i, S, S_i \in \mathsf{roles}(\mathfrak{R}(C))$, $w, w_i \in \mathsf{roles}(\mathfrak{R})^+$, we have
   (a) if $C_1 \in \mathcal{L}_{\mathcal{N}}(x)$, then $\neg C_1 \notin \mathcal{L}_{\mathcal{N}}(x)$,
   (b) if $C_1 \sqcap C_2 \in \mathcal{L}_{\mathcal{N}}(x)$, then $C_1 \in \mathcal{L}_{\mathcal{N}}(x)$ and $C_2 \in \mathcal{L}_{\mathcal{N}}(x)$,
   (c) if $C_1 \sqcup C_2 \in \mathcal{L}_{\mathcal{N}}(x)$, then $C_1 \in \mathcal{L}_{\mathcal{N}}(x)$ or $C_2 \in \mathcal{L}_{\mathcal{N}}(x)$,
   (d) if $\exists R.C_1 \in \mathcal{L}_{\mathcal{N}}(x)$, then there is some y such that $\langle x, y \rangle \in \mathcal{E}_R$ and $C_1 \in \mathcal{L}_{\mathcal{N}}(y)$,
   (e) if $(\forall R.C_1)_{R,w} \in \mathcal{L}_{\mathcal{N}}(x)$, then $C_1 \in \mathcal{L}_{\mathcal{N}}(x)$,
   (f) $\forall R_?.\top \in \mathcal{L}_{\mathcal{N}}(x)$,
   (g) $(\forall R.C_1)_{S,w} \in \mathcal{L}_{\mathcal{N}}(y)$ iff there is some x with $x = \mathsf{w\_ancestor}(y, w)$ and $\forall R.C_1 \in \mathcal{L}_{\mathcal{N}}(x)$,
   (h) if $(\forall R.C_1)_{S,w} \in \mathcal{L}_{\mathcal{N}}(x)$ and $|w| = 1$, then $w = S$,
   (i) if $(\forall R_1.C_1)_{S_1,w} \in \mathcal{L}_{\mathcal{N}}(x)$ and $(\forall R_2.C_2)_{S_2,w} \in \mathcal{L}_{\mathcal{N}}(x)$, then $S_1 = S_2$,
   (j) if $(\forall R_1.C_1)_{S_1,w_1} \in \mathcal{L}_{\mathcal{N}}(x)$, $(\forall R_2.C_2)_{S_2,w_2} \in \mathcal{L}_{\mathcal{N}}(y)$, $(\forall R_3.C_3)_{S_3,w_1 w_2} \in \mathcal{L}_{\mathcal{N}}(y)$ and $x = \mathsf{w\_ancestor}(y, w_2)$, then $S_3 \in \mathsf{con}(S_1, S_2)$.

Lemma 1 $(C, \mathfrak{R})$ is satisfiable iff there exists a tableau $\mathcal{T}$ for $(\mathsf{NNF}(C), \mathfrak{R}(C))$.

Proof 2 Due to Proposition 2, $(C,\mathfrak{R})$ is satisfiable iff $(\mathsf{NNF}(C),\mathfrak{R})$ is satisfiable. Due to Proposition 3, $(\mathsf{NNF}(C),\mathfrak{R})$ is satisfiable iff $(\mathsf{NNF}(C),\mathfrak{R}(C))$ is satisfiable. Let $C' = \mathsf{NNF}(C)$.

"$\Leftarrow$": If $\mathcal{T} = (\mathcal{N},\mathcal{L}_\mathcal{N},\mathcal{E},\mathcal{L}_\mathcal{E})$ is a tableau for $(C',\mathfrak{R}(C'))$, a model $\mathcal{I} = (\Delta^\mathcal{I},\cdot^\mathcal{I})$ of $(C',\mathfrak{R}(C'))$ can be constructed as follows: $\Delta^\mathcal{I} =_{def} \mathcal{N}$, $C_1^\mathcal{I} =_{def} \{\, x \mid C_1 \in \mathcal{L}_\mathcal{N}(x)\,\}$ for all $C_1 \in \mathcal{N}_\mathcal{C} \cap \mathsf{sub}(C')$, $C_1^\mathcal{I} =_{def} \emptyset$ for all other concept names $C_1 \in \mathcal{N}_\mathcal{C} \setminus \mathsf{sub}(C')$, and
$R^\mathcal{I} =_{def} \{\, \langle x,y\rangle \mid \mathsf{w\_ancestor}(y,w) = x, (\forall S.C_1)_{R,w} \in \mathcal{L}_\mathcal{N}(y)\,\}$ for all role names $R \in \mathsf{roles}(\mathfrak{R}(C))$, and $R^\mathcal{I} =_{def} \emptyset$ for all other roles $R \in \mathcal{N}_\mathcal{R} \setminus \mathsf{roles}(\mathfrak{R}(C))$. First of all, due to Proposition 1 we can safely interpret all unmentioned roles (those not in $\mathsf{roles}(\mathfrak{R}(C'))$) and concept names (those not in $\mathsf{sub}(C')$) with the empty set. We show that all roles are interpreted as disjoint: assume the contrary. Then there must be some roles $R, S \in \mathcal{N}_\mathcal{R}$, $R \neq S$: $R^\mathcal{I} \cap S^\mathcal{I} \neq \emptyset$. Due to the definition of $\cdot^\mathcal{I}$, $\langle x,y\rangle \in R^\mathcal{I} \cap S^\mathcal{I}$ iff $x = \mathsf{w\_ancestor}(y,w)$, $(\forall S_1.C_1)_{R,w} \in \mathcal{L}_\mathcal{N}(y)$ and $(\forall S_2.C_2)_{S,w} \in \mathcal{L}_\mathcal{N}(y)$. However, this violates Property 2i, and therefore, $\mathcal{T}$ cannot be a tableau (contradiction). We show $\mathcal{I} \models \mathfrak{R}(C')$: assume the contrary. Then there must be some role axiom $ra \in \mathfrak{R}(C')$ that ist not satisfied by $\mathcal{I}$. This is the case iff there are some x,y,z with $\langle x,y\rangle \in R^\mathcal{I}$, $\langle y,z\rangle \in S^\mathcal{I}$, and either $\langle x,z\rangle \notin \mathcal{UR}(\mathcal{I})$, or $\langle x,z\rangle \in T^\mathcal{I}$, but $T \notin \mathsf{con}(R,S)$. Due to the definition of $\cdot^\mathcal{I}$, $\langle x,y\rangle \in R^\mathcal{I}$, $\langle y,z\rangle \in S^\mathcal{I}$, iff $x = \mathsf{w\_ancestor}(y,w_1)$, $y = \mathsf{w\_ancestor}(z,w_2)$, $(\forall S_1.C_1)_{R,w_1} \in \mathcal{L}_\mathcal{N}(y)$ and $(\forall S_2.C_2)_{S,w_2} \in \mathcal{L}_\mathcal{N}(z)$. In the first case, $\langle x,z\rangle \notin \mathcal{UR}(\mathcal{I})$, iff $(\forall S_3.C_3)_{T,w_1w_2} \notin \mathcal{L}_\mathcal{N}(z)$, for all $T \in \mathcal{N}_\mathcal{R}$. However, due to property 2f, we have $\forall R_?.\top \in \mathcal{L}_\mathcal{N}(x)$. Since $x = \mathsf{w\_ancestor}(z,w)$ and $\forall R_?.\top \in \mathcal{L}_\mathcal{N}(x)$, due to property 2g we have $(\forall R_?.\top)_{T,w} \in \mathcal{L}_\mathcal{N}(z)$, for some $T \in \mathsf{roles}(\mathfrak{R}(C'))$ (contradiction). In the second case, $\langle x,z\rangle \in T^\mathcal{I}$ iff $(\forall S_3.C_3)_{T,w_1w_2} \in \mathcal{L}_\mathcal{N}(z)$. Summing up we have $y = \mathsf{w\_ancestor}(z,w_2)$, $(\forall S_1.C_1)_{R,w_1} \in \mathcal{L}_\mathcal{N}(y)$ and $(\forall S_2.C_2)_{S,w_2} \in \mathcal{L}_\mathcal{N}(z)$. $(\forall S_3.C_3)_{T,w_1w_2} \in \mathcal{L}_\mathcal{N}(z)$. Due to tableau Condition 2j, $T \in \mathsf{con}(R,S)$ (contradiction). We can also show by structural induction on the concept E that if $E \in \mathcal{L}_\mathcal{N}(x)$, then also $x \in E^\mathcal{I}$ (see [3] for details). Assuming this, from $C' \in \mathcal{L}_\mathcal{N}(x_0)$ (due to Property 1 in Definition 3) it follows that $x_0 \in C'^\mathcal{I}$. Since $C'^\mathcal{I} \neq \emptyset$, we have $\mathcal{I} \models C'$. Summing up, we have shown that $\mathcal{I} \models (C',\mathfrak{R}(C'))$.

"$\Rightarrow$": If $\mathcal{I} \models (C',\mathfrak{R}(C'))$, $\mathcal{I} = (\Delta^\mathcal{I},\cdot^\mathcal{I})$, then a tableau $\mathcal{T} = (\mathcal{N},\mathcal{L}_\mathcal{N},\mathcal{E},\mathcal{L}_\mathcal{E})$ for $(C',\mathfrak{R}(C'))$ can be constructed. Since a tableau is required to be a (possibly infinite) tree, but a model may contain "joins" and cycles and is therefore an arbitrary graph, but not necessarily a tree, we cannot simply assign $\mathcal{N} =_{def} \Delta^\mathcal{I}$. Intuitively, the tableau is constructed by *traversing* the model, collecting the required information. Therefore, each node $x \in \mathcal{N}$ in the tableau $\mathcal{T}$ corresponds to a *path* in $\mathcal{I}$. A path in $\mathcal{I}$ is inductively defined as follows:

- for some (but only one) $i_0 \in \Delta^\mathcal{I}$ with $i_0 \in C'^\mathcal{I}$, $[i_0]$ is a path in $\mathcal{I}$
- if $[i_0,\ldots,i_m]$ is a path in $\mathcal{I}$ and $i_m \in (\exists R.C_1)^\mathcal{I}$, $\langle i_m,i_n\rangle \in R^\mathcal{I}$ with $i_n \in C_1^\mathcal{I}$ for some $\exists R.C_1 \in \mathsf{sub}(C')$, then $[i_0,\ldots,i_m,i_n]$ is also a path in $\mathcal{I}$.

Let $\mathcal{P}(\mathcal{I})$ denote the set of paths (as defined above) in $\mathcal{I}$. We can now define $\mathcal{T} = (\mathcal{N}, \mathcal{L}_\mathcal{N}, \mathcal{E}, \mathcal{L}_\mathcal{E})$ as follows

- $\mathcal{N} =_{def} \mathcal{P}(\mathcal{I})$,
- $\mathcal{E} =_{def} \{ <p,q> \mid p, q \in \mathcal{N}, p = [i_0, \ldots, i_n], q = [i_0, \ldots, i_n, i_{n+1}], \text{ (possibly } n = 0), <i_n, i_{n+1}> \in \mathcal{UR}(\mathcal{I}, \mathsf{roles}(\mathfrak{R}(C'))) \}$,
- $\mathcal{L}_\mathcal{E} =_{def} \{ (<p,q>, R) \mid <p,q> \in \mathcal{E}, p = [i_0, \ldots, i_n], q = [i_0, \ldots, i_n, i_{n+1}], \text{ (possibly } n = 0), <i_n, i_{n+1}> \in R^\mathcal{I} \}$,
- For all $q \in \mathcal{N}$, $q = [i_0, \ldots, i_n]$:
  $\mathcal{L}_\mathcal{N}(q) =_{def} \{ C_1 \mid C_1 \in \mathsf{sub}(C'), i_n \in C_1^\mathcal{I} \} \cup \{\forall R_?.\top\} \cup \{ (\forall R.C_1)_{S,w} \mid p = \mathsf{w\_ancestor}(q,w), p = [i_0, \ldots, i_m], q = [i_0, \ldots, i_m, \ldots, i_n], <i_m, i_n> \in S^\mathcal{I}, \forall R.C_1 \in \mathcal{L}_\mathcal{N}(p) \}$.

We have to prove that $\mathcal{T}$ is a tableau for $(C', \mathfrak{R}(C'))$ by showing that the tableau conditions are satisfied. First of all, $(\mathcal{N}, \mathcal{E})$ is indeed a (possibly infinite) tree which should be obvious by the definitions of $\mathcal{N}$ and $\mathcal{E}$. Then, it can be shown that the construction satisfies the tableau properties 1 to 2j (see [3] for details). Summing up we have shown that $\mathcal{T}$ is a tableau for $(C', \mathfrak{R}(C'))$. □

Definition 4 (Completion Tree) A completion tree $\mathcal{CT}$ for $(C, \mathfrak{R}(C))$ is a tuple $(\mathcal{N}, \mathcal{L}_\mathcal{N}, \mathcal{E}, \mathcal{L}_\mathcal{E})$. $\mathcal{N}$, $\mathcal{E}$, $\mathcal{L}_\mathcal{N}$ and $\mathcal{L}_\mathcal{E}$ are defined as in Definition 3, but without the additional conditions 1 and 2a – 2j. Unlike in Definition 3, $(\mathcal{N}, \mathcal{E})$ is always a *finite* tree. The same notions of successor, w_ancestor etc. as in Definition 3 are used. A completion tree is said to contain a *clash* iff there is some node x with $\{C, \neg C\} \subseteq \mathcal{L}_\mathcal{N}(x)$ for some $C \in \mathcal{N}_\mathcal{C}$ (*primitive clash*), or there are constraints $(\forall R_1.C_1)_{S,w_1} \in \mathcal{L}_\mathcal{N}(x)$, $(\forall R_2.C_2)_{T,w_2} \in \mathcal{L}_\mathcal{N}(y)$, $(\forall R_3.C_3)_{U,w_1 w_2} \in \mathcal{L}_\mathcal{N}(y)$ with $x = \mathsf{w\_ancestor}(y, w_2)$ and $U \notin \mathsf{con}(S,T)$ (*role box clash*). Two nodes x, y in a completion tree are said to be *equivalent*, $x \equiv y$ iff $\forall c_1 : (c_1 \in \mathcal{L}_\mathcal{N}(x) \Rightarrow \exists c_2 \in \mathcal{L}_\mathcal{N}(y) : c_1 \equiv_c c_2) \wedge \forall c_1 : (c_1 \in \mathcal{L}_\mathcal{N}(y) \Rightarrow \exists c_2 \in \mathcal{L}_\mathcal{N}(x) : c_1 \equiv_c c_2)$, where $c_1 \equiv_c c_2$ iff $c_1 = (\forall R.C)_{S,w} \wedge c_2 = (\forall R.C)_{S,v} \vee c_1 = c_2$.

The tableaux algorithm works as follows: in order to decide the satisfiability of $(C, \mathfrak{R}(C))$, the algorithm starts with the initial completion tree

$$\mathcal{CT}_0 = (\{x_0\}, \{(x_0, \{\mathsf{NNF}(C) \sqcap \forall R_?.\top\})\}, \emptyset, \emptyset)$$

and exhaustively applies the non-deterministic *tableaux expansion rules* (see Figure 1) until either the completion tree contains a clash (see above), or none of the rules can be applied any longer, i.e. the tree is complete. If the completion rules can be applied in such a way that they construct a complete and clash-free completion tree, then $(C, \mathfrak{R}(C))$ is satisfiable. Then, $(C, \mathfrak{R}(C))$ is unsatisfiable iff *all* possible computations yield a completion tree containing a clash.

$\sqcap$-rule:
if 1. $C_1 \sqcap C_2 \in \mathcal{L}_\mathcal{N}(x_i)$
2. $\{C_1, C_2\} \not\subseteq \mathcal{L}_\mathcal{N}(x_i)$
then $\mathcal{L}_\mathcal{N}(x_i) := \mathcal{L}_\mathcal{N}(x_i) \cup \{C_1, C_2\}$

$\sqcup$-rule:
if 1. $C_1 \sqcup C_2 \in \mathcal{L}_\mathcal{N}(x_i)$
2. $\{C_1, C_2\} \cap \mathcal{L}_\mathcal{N}(x_i) = \emptyset$
then
$\mathcal{L}_\mathcal{N}(x_i) := \mathcal{L}_\mathcal{N}(x_i) \cup \{C\}$
for some $C \in \{C_1, C_2\}$

$\forall$-rule:
if 1. $(\forall R.D)_{R,w} \in \mathcal{L}_\mathcal{N}(x_i)$
2. $D \notin \mathcal{L}_\mathcal{N}(x_i)$
then
$\mathcal{L}_\mathcal{N}(x_i) := \mathcal{L}_\mathcal{N}(x_i) \cup \{D\}$

$\exists\forall$-rule:
if 1. $\exists R.C_1 \in \mathcal{L}_\mathcal{N}(x_i)$
2. neither the $\sqcap$- nor the $\sqcup$- nor the $\forall$-rule is applicable to x_i
3. $\neg\exists \langle x_i, x_j\rangle \in \mathcal{E}_\mathcal{R} : C_1 \in \mathcal{L}_\mathcal{N}(x_j)$
4. x_i is not **blocked** (see below for a discussion)
then create a new node x_j with
$\mathcal{L}_\mathcal{E}(\langle x_i, x_j\rangle) := R$, $\mathcal{L}_\mathcal{N}(x_j) := \mathcal{L}$,
where $\mathcal{W} = \{\, (w, S) \mid (\forall T.D)_{S,w} \in \mathcal{L}_\mathcal{N}(x_i) \,\}$,
and $\mathcal{L}$ is some set that can non-deterministically be constructed by:
for all $(w, S) \in \mathcal{W}$:
choose some $U \in \mathsf{con}(S, R)$:
$\mathcal{C}(w) = \{\, (\forall T.D)_{U,wR} \mid (\forall T.D)_{S,w} \in \mathcal{L}_\mathcal{N}(x_i) \,\}$
$\mathcal{L} = \{C_1\} \cup \{\forall R_?.\top\} \cup \bigcup_{(w,S)\in\mathcal{W}} \mathcal{C}(w) \cup \{\, (\forall T.D)_{R,R} \mid \forall T.D \in \mathcal{L}_\mathcal{N}(x_i) \,\}$

Figure 1: The tableaux expansion rules

A *blocking mechanism* is needed to ensure termination of the tableaux algorithm, e.g. for $((\exists R.C) \sqcap (\forall R.\exists R.C), \{R \circ R \sqsubseteq R\})$. Note that this concept is also expressible in $\mathcal{ALC}_{\mathcal{R}^+}$, since R is declared as a transitively closed role. Unlike for $\mathcal{ALC}_{\mathcal{R}^+}$, an infinite model (tableau) must be constructed if blocking occurred. Unfortunately, the blocking condition for $\mathcal{ALC}_{\mathcal{RA}}$ is not yet thoroughly worked out, so we will discuss some open problems in the following. However, we strongly believe that an appropriate blocking condition can be found and that the language is therefore indeed decidable, even if no formal proof is yet available. Surprisingly, unlike for $\mathcal{ALC}_{\mathcal{R}^+}$, *equal blocking* is not correct. Considering equal blocking, a node y is said to be blocked by an ancestor node x of y, iff $\mathcal{L}_\mathcal{N}(x) \equiv \mathcal{L}_\mathcal{N}(y)$. Note that $(\forall R.C)_{S,w_1} \equiv_c (\forall R.C)_{S,w_2}$, even if $w_1 \neq w_2$. In order to exemplify the incorrectness of equal blocking, let us consider

$$(\exists R.\top \sqcap \forall R.\exists R.\top \sqcap \forall S.\exists R.\top, \{R \circ R \sqsubseteq S, S \circ R \sqsubseteq S, S \circ S \sqsubseteq T\}).$$

The example is unsatisfiable, since a *role box clash* is encountered if a chain of at least four R successors ($\langle x_0, x_1\rangle \in \mathcal{E}_\mathcal{R}$, $\langle x_1, x_2\rangle \in \mathcal{E}_\mathcal{R}$, $\langle x_2, x_3\rangle \in \mathcal{E}_\mathcal{R}$, $\langle x_3, x_4\rangle \in \mathcal{E}_\mathcal{R}$) has been created:
$\mathcal{L}_\mathcal{N}(x_0) = \{\exists R.\top \sqcap \forall R.\exists R.\top \sqcap \forall S.\exists R.\top \sqcap \forall R_?.\top, \exists R.\top, \forall R.\exists R.\top, \forall S.\exists R.\top, \forall R_?.\top\}$,
$\mathcal{L}_\mathcal{N}(x_1) = \{\top, \exists R.\top, (\forall R.\exists R.\top)_{R,R}, (\forall S.\exists R.\top)_{R,R}, \forall R_?.\top, (\forall R_?.\top)_{R,R}\}$,
$\mathcal{L}_\mathcal{N}(x_2) = \{\top, \exists R.\top, (\forall R.\exists R.\top)_{S,R^2}, (\forall S.\exists R.\top)_{S,R^2}, \forall R_?.\top, (\forall R_?.\top)_{R,R}, (\forall R_?.\top)_{S,R^2}\}$,
$\mathcal{L}_\mathcal{N}(x_3) = \{\top, \exists R.\top, (\forall R.\exists R.\top)_{S,R^3}, (\forall S.\exists R.\top)_{S,R^3}, \forall R_?.\top, (\forall R_?.\top)_{R,R}, (\forall R_?.\top)_{S,R^2}, (\forall R_?.\top)_{S,R^3}\}$, $\mathcal{L}_\mathcal{N}(x_4) = \{\top, \exists R.\top, (\forall R.\exists R.\top)_{S,R^4}, (\forall S.\exists R.\top)_{S,R^4}, \forall R_?.\top, (\forall R_?.\top)_{R,R},$

$(\forall R_?.\top)_{S,R^2}, (\forall R_?.\top)_{S,R^3}, (\forall R_?.\top)_{S,R^4}\}$. The completion tree contains a role box clash due to $(\forall R_?.\top)_{S,R^4} \in \mathcal{L}_\mathcal{N}(x_4)$, $(\forall R_?.\top)_{S,R^2} \in \mathcal{L}_\mathcal{N}(x_4)$, $(\forall R_?.\top)_{S,R^2} \in \mathcal{L}_\mathcal{N}(x_2)$, with $x_2 = \mathsf{w\_ancestor}(x_4, RR)$ and $S \notin \mathsf{con}(S,S)$. Considering equal blocking, node x_3 would have already been blocked by x_2 since $\mathcal{L}_\mathcal{N}(x_3) \equiv \mathcal{L}_\mathcal{N}(x_2)$, and the wrong answer "satisfiable" would be returned by the algorithm.

A promising candidate for a blocking condition is the predicate $\mathcal{L}_\mathcal{N}(x) \equiv_c \mathcal{L}_\mathcal{N}(y) \wedge \mathcal{COMP}(x) = \mathcal{COMP}(y)$, where $\mathcal{COMP}(z)$ is the *set of compositions* for the node z: $\mathcal{COMP}(z) =_{def} \{(S,T,U) \mid \exists w_1, w_2, x' : \mathsf{w\_ancestor}(z, w_2) = z'$, $(\forall S_1.C_1)_{S,w1} \in \mathcal{L}_\mathcal{N}(z'), (\forall S_2.C_2)_{T,w2} \in \mathcal{L}_\mathcal{N}(z), (\forall S_3.C_3)_{U,w1w2} \in \mathcal{L}_\mathcal{N}(z)\}$. Reconsidering our example, w.r.t. this new blocking condition x_3 is not blocked by x_2, since $\mathcal{COMP}(x_2) \neq \mathcal{COMP}(x_3)$: $\mathcal{COMP}(x_2) = \{(R,R,S)\}$, $\mathcal{COMP}(x_3) = \{(R,R,S),(S,R,S)\}$. Therefore, x_4 would have been created, and the unsatisfiability due to the role box clash would be detected. Intuitively, $\mathcal{COMP}(x) = \mathcal{COMP}(y)$ ensures that no new composition possibilities have been produced that might lead to role box clashes when extending the tableau.

Let us discuss why a *complete role box* is needed. Let us consider $(\forall V.C) \sqcap (\exists R.\exists S.\exists T.\neg C)$ w.r.t. $\{S \circ T \sqsubseteq U, R \circ U \sqsubseteq V\}$, which is unsatisfiable. Assume the algorithm would be run *without* a completed role box, and let $\langle x_0, x_1 \rangle \in \mathcal{E}_\mathcal{R}$, $\langle x_1, x_2 \rangle \in \mathcal{E}_\mathcal{S}$, $\langle x_2, x_3 \rangle \in \mathcal{E}_\mathcal{T}$. Then, $\forall V.C \in \mathcal{L}_\mathcal{N}(x_0)$, $(\forall V.C)_{R,R} \in \mathcal{L}_\mathcal{N}(x_1)$, but $(\forall V.C)_{X,RS} \notin \mathcal{L}_\mathcal{N}(x_2)$ for all roles X, since there is no role axiom with $\mathsf{pre}(pra) = (R,S)$. Therefore, the qualification $\forall V.C$ was "forgotten" and no clash would be detected. However, the completed role box contains $R \circ S \sqsubseteq R_? \sqcup R \sqcup S \sqcup T \sqcup U \sqcup V$. Therefore, $(\forall V.C)_{X,RS} \in \mathcal{L}_\mathcal{N}(x_2)$ for some $X \in \{R_?, R, S, T, U, V\}$. The constraint will be propagated to x_3 as $(\forall V.C)_{Y,RST} \in \mathcal{L}_\mathcal{N}(x_3)$ for some $Y \in \mathsf{con}(X,T)$. Since the role box is complete, $Y \in \mathsf{con}(X,T)$ can indeed be choosen, independently of X. Now, due to $\forall R_?.\top \in \mathcal{L}_\mathcal{N}(x_1)$ and $S \circ T \sqsubseteq U$, we also have $(\forall R_?.\top)_{S,S} \in \mathcal{L}_\mathcal{N}(x_2)$, $(\forall R_?.\top)_{U,ST} \in \mathcal{L}_\mathcal{N}(x_3)$. Since $(\forall V.C)_{R,R} \in \mathcal{L}_\mathcal{N}(x_1)$, $(\forall R_?.\top)_{U,ST} \in \mathcal{L}_\mathcal{N}(x_3)$ and $\mathsf{con}(R,U) = V$, a role box clash can only be avoided iff $(\forall V.C)_{Y,RST} = (\forall V.C)_{V,RST}$, i.e. $Y = V$. In fact, $Y = V$ must have been chosen in order to avoid the role box clash, so $(\forall V.C)_{V,RST} \in \mathcal{L}_\mathcal{N}(x_3)$. However, this produces a primitive clash, as desired, due to $\{\neg C, C\} \subseteq \mathcal{L}_\mathcal{N}(x_3)$. Whatever is tried (guessed) for X and Y, either a role box clash or a primitive clash is detected, proving the unsatisfiability of the example. Note that the completed role box as well as the (appropriately rewritten) $\forall R_?.\top$ constraints play a key role in this argumentation. Unfortunately, it is *not* straightforward to extend the calculus to be able to deal with non-disjoint roles. In fact, the disjointness requirement is crucial for the approach. Let us consider $(\forall B.C) \sqcap (\forall V.\bot) \sqcap \exists R.((\forall U'.\bot) \sqcap \exists S.\exists T.\neg C)$ w.r.t. the role box $\{R \circ S \sqsubseteq U \sqcup V, S \circ T \sqsubseteq U' \sqcup V', R \circ U' \sqsubseteq A, R \circ V' \sqsubseteq B, U \circ T \sqsubseteq A, V \circ T \sqsubseteq B\}$. Then, the following computation is possible: $(\forall V.\bot)_{U,RS} \in \mathcal{L}_\mathcal{N}(x_2)$, $(\forall B.C)_{U,RS} \in \mathcal{L}_\mathcal{N}(x_2)$ (due to $R \circ S \sqsubseteq U \sqcup V$ and $\forall V.\bot \in \mathcal{L}_\mathcal{N}(x_0)$), $(\forall V.\bot)_{A,RST} \in \mathcal{L}_\mathcal{N}(x_3)$, $(\forall B.C)_{A,RST} \in \mathcal{L}_\mathcal{N}(x_3)$ (due to $U \circ T \sqsubseteq A$). Since $\exists T.\neg C \in \mathcal{L}_\mathcal{N}(x_2)$, $\neg C \in \mathcal{L}_\mathcal{N}(x_3)$. Additionally, $C \notin \mathcal{L}_\mathcal{N}(x_3)$, since the qualification $(\forall B.C)_{A,RST} \in \mathcal{L}_{\mathcal{N}}(x_3)$ is not applicable, because $B \neq A$. Due to $\forall U'.\bot \in \mathcal{L}_{\mathcal{N}}(x_1)$ and $S \circ T \sqsubseteq U' \sqcup V'$, we also have $(\forall U'.\bot)_{V',ST} \in \mathcal{L}_{\mathcal{N}}(x_3)$. Now, due to $R \circ V' \sqsubseteq B$ the qualification C from $\forall B.C \in \mathcal{L}_{\mathcal{N}}(x_0)$ should be added to $\mathcal{L}_{\mathcal{N}}(x_3)$, since x_3 is an indirect A-successor *and* an indirect B-successor of x_0. With the disjointness requirement, we would get a role box clash, since $A \in \mathsf{con}(U,T)$ has been choosen, but $A \notin \mathsf{con}(R,V')$. But without the disjointness requirement, no clash is detected because $(\forall B.C)_{B,RST} \notin \mathcal{L}_{\mathcal{N}}(x_3)$. This is due to the fact that the calculus creates only "left reductions" when propagating the annotated all constraints to the next successor node ($(R \circ S) \circ T$ is called *left reduction*, $R \circ (S \circ T)$ *right reduction*). The disjointness requirement enforces that the choosen left reduction is in fact the *only* valid reduction possibility. As the example shows, without the disjointness requirement, other reduction possibilities could enforce non-empty role intersections without being noticed, since there is no "syntactic indicator" for them (as for B in the example).

4 Conclusion and Future Work

It has been argued that the satisfiability problem of the language $\mathcal{ALC}_{\mathcal{RA}}$ is decidable. It should be noticed that the so-called *trace technique* can be applied for $\mathcal{ALC}_{\mathcal{RA}}$, but we were not able to establish a polynomial bound on the length of the traces. It is believed that the satisfiability problem of $\mathcal{ALC}_{\mathcal{RA}}$ is not in PSPACE. We are currently working on a calculus for a language called $\mathcal{ALCH}_{\mathcal{RA}^\ominus}$. This language does not require that all roles have to be interpreted as disjoint, and also role inclusion axioms ($R \sqsubseteq S$) are allowed. Therefore, $\mathcal{ALCH}_{\mathcal{RA}^\ominus}$ is a full super-language of $\mathcal{ALCH}_{\mathcal{R}^+}$. Since the presented tableaux-based approach cannot be extended to cover non-disjoint roles correctly, we are trying to represent the direct as well as the indirect edges explicitly via role membership constraints (constraints of the form $(x,y) : R$). Blocking seems to be even more complicated with this approach. However, $\mathcal{ALCH}_{\mathcal{RA}^\ominus}$ has the finite model property, unlike $\mathcal{ALC}_{\mathcal{RA}}$. Finally, we would like to thank the anonymous reviewers and Anni-Yasmin Turhan for valuable comments.

References

[1] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical reasoning for expressive description logics. In *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99).*

[2] Manfred Schmidt-Schauß. Subsumption in KL-ONE is Undecidable. In *Principle of Knowledge Representation and Reasoning – Proceedings of the First International Conference KR '89.*

[3] M. Wessel, V. Haarslev, and R. Möller. $\mathcal{ALC}_{\mathcal{RA}}$ – $\mathcal{ALC}$ with Role Axioms. Forthcoming. Technical report.