RWTH
LTCS–Report

Aachen University of Technology
Research group for
Theoretical Computer Science

A Description Logic with Transitive and Converse Roles and Role Hierarchies

Ian Horrocks and Ulrike Sattler

LTCS-Report 98-05

An abridged version will appear in the Proceedings of the International Workshop on Description Logics, Trento, Italy, 1998.

RWTH Aachen
LuFg Theoretische Informatik
http://www-lti.informatik.rwth-aachen.de

Ahornstr. 55
52074 Aachen
Germany

A Description Logic with Transitive and Converse Roles and Role Hierarchies

Ian Horrocks and Ulrike Sattler

1 Motivation

As widely argued [Horrocks & Gough,1997; Sattler,1996], transitive roles play an important rôle in the adequate representation of aggregated objects: they allow these objects to be described by referring to their parts without specifying a level of decomposition. In [Horrocks & Gough,1997], the Description Logic (DL) $\mathcal{ALCH}_{R^+}$ is presented, which extends $\mathcal{ALC}$ with transitive roles and a role hierarchy. It is argued in [Sattler,1998] that $\mathcal{ALCH}_{R^+}$ is well-suited to the representation of aggregated objects in applications that require various part-whole relations to be distinguished, some of which are transitive. However, $\mathcal{ALCH}_{R^+}$ allows neither the description of parts by means of the whole to which they belong, or vice versa. To overcome this limitation, we present the DL $\mathcal{ALCHI}_{R^+}$ which allows the use of, for example, `has_part` as well as `is_part_of`. To achieve this, $\mathcal{ALCH}_{R^+}$ was extended with inverse roles.

It could be argued that, instead of defining yet another DL, one could make use of the results presented in [De Giacomo & Lenzerini,1996] and use $\mathcal{ALC}$ extended with role expressions which include transitive closure and inverse operators. The reason for not proceeding like this is the fact that transitive roles can be implemented more efficiently than the transitive closure of roles (see [Horrocks & Gough,1997]), although they lead to the same complexity class (ExpTime-hard) when added, together with role hierarchies, to $\mathcal{ALC}$. Furthermore, it is still an open question whether the transitive closure

of roles together with inverse roles necessitates the use of the cut rule [De Giacomo & Massacci,1998], and this rule leads to an algorithm with very bad behaviour. We will present an algorithm for $\mathcal{ALCHI}_{R^+}$ without such a rule.[1]

2 A Tableaux Algorithm for $\mathcal{ALCI}_{R^+}$

In this section a tableaux algorithm for testing the satisfiability of $\mathcal{ALCI}_{R^+}$ concept expressions will be described and a proof of its soundness and completeness presented. The algorithm and proof are extensions of those described for $\mathcal{ALC}_{R^+}$ [Sattler,1996].

2.1 Syntax and Semantics

$\mathcal{ALCI}_{R^+}$ is the Description Logic (DL) obtained by augmenting the well-known DL $\mathcal{ALC}$ [Schmidt-Schauß & Smolka,1988] with *transitively closed roles* and *inverse* (converse) roles. The set of transitive role names $\mathbf{R}_+$ is a subset of the set of role names $\mathbf{R}$. Interpretations map role names to binary relations on the interpretation domain, and transitive role names to transitive relations. In addition, for any role $R \in \mathbf{R}$, the role R^- is interpreted as the inverse of R.

Definition 1 Let N_C be a set of *concept names* and let $\mathbf{R}$ be a set of *role names* with both transitive and normal role names $\mathbf{R}_+ \cup \mathbf{R}_\mathsf{P} = \mathbf{R}$, where $\mathbf{R}_\mathsf{P} \cap \mathbf{R}_+ = \emptyset$. The set of $\mathcal{ALCI}_{R^+}$-*roles* is $\mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$. The set of $\mathcal{ALCI}_{R^+}$-*concepts* is the smallest set such that

1. every concept name $C \in N_C$ is a concept and
2. if C and D are concepts and R is an $\mathcal{ALCI}_{R^+}$-role, then $(C \sqcap D)$, $(C \sqcup D)$, $(\neg C)$, $(\forall R.C)$, and $(\exists R.C)$ are concepts.

An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of a set $\Delta^\mathcal{I}$, called the *domain* of $\mathcal{I}$, and a function $\cdot^\mathcal{I}$ which maps every concept to a subset of $\Delta^\mathcal{I}$ and every role to a subset of $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$ such that

$$(C \sqcap D)^\mathcal{I} \;=\; C^\mathcal{I} \cap D^\mathcal{I},$$

[1] Many details, and the proofs of the various lemmata, have been omitted in the interests of brevity.

$$
\begin{aligned}
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}}, \\
\neg C^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, \\
(\exists R.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \text{There exists some } y \in \Delta^{\mathcal{I}} \text{ with } \langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}, \\
(\forall R.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \text{For all } y \in \Delta^{\mathcal{I}}, \text{ if } \langle x, y\rangle \in R^{\mathcal{I}}, \text{ then } y \in C^{\mathcal{I}}\},
\end{aligned}
$$

and, for $P \in \mathbf{R}$ and $R \in \mathbf{R}_+$,

$$
\begin{gathered}
\langle x, y\rangle \in P^{\mathcal{I}} \text{ iff } \langle y, x\rangle \in P^{-\mathcal{I}}, \\
\text{if } \langle x, y\rangle \in R^{\mathcal{I}} \text{ and } \langle y, z\rangle \in R^{\mathcal{I}}, \text{ then } \langle x, z\rangle \in R^{\mathcal{I}}.
\end{gathered}
$$

A concept C is called *satisfiable* iff there is some interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$. Such an interpretation is called a *model of* C. A concept D *subsumes* a concept C (written $C \sqsubseteq D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for each interpretation $\mathcal{I}$. Two concepts C, D are *equivalent* (written $C \equiv D$) iff they are mutually subsuming. For an interpretation $\mathcal{I}$, an individual $x \in \Delta^{\mathcal{I}}$ is called an *instance* of a concept C iff $x \in C^{\mathcal{I}}$.

In order to make the following considerations easier, we introduce two functions on roles:

1. The inverse relation on roles is symmetric, and to avoid considering roles such as R^{--}, we define a function Inv which returns the inverse of a role, more precisely

$$
\mathsf{Inv}(R) := \begin{cases} R^- & \text{if } R \text{ is a role name,} \\ S & \text{if } R = S^- \text{ for a role name } S. \end{cases}
$$

2. Obviously, a role R is transitive if and only if $\mathsf{Inv}(R)$ is transitive. However, either R or $\mathsf{Inv}(R)$ is in $\mathbf{R}_+$. In order to avoid this case distinction, the function Trans returns true iff R is a transitive role—regardless whether it is a role name or the inverse of a role name.

$$
\mathsf{Trans}(R) := \begin{cases} \text{true} & \text{if } R \in \mathbf{R}_+ \text{ or } \mathsf{Inv}(R) \in \mathbf{R}_+, \\ \text{false} & \text{otherwise.} \end{cases}
$$

2.2 An $\mathcal{ALCI}_{R^+}$ Tableau

Like other tableaux algorithms, the $\mathcal{ALCI}_{R^+}$ algorithm tries to prove the satisfiability of a concept expression D by constructing a model of D. The

model is represented by a so-called *completion tree*, a tree some of whose nodes correspond to individuals in the model, each node being labelled with a set of $\mathcal{ALCI}_{R^+}$-concepts. When testing the satisfiability of an $\mathcal{ALCI}_{R^+}$-concept D, these sets are restricted to subsets of $sub(D)$, where $sub(D)$ is the set of subconcepts of D.

For ease of construction, we assume all concepts to be in *negation normal form* (NNF), that is, negation occurs only in front of concept names. Any $\mathcal{ALCI}_{R^+}$-concept can easily be extended to an equivalent one in NNF by pushing negations inwards using a combination of DeMorgan's laws and the following equivalences:

$$\begin{aligned} \neg(C \sqcup D) &\equiv \neg C \sqcap \neg D \\ \neg(C \sqcap D) &\equiv \neg C \sqcup \neg D \\ \neg(\exists R.C) &\equiv (\forall R.\neg C) \\ \neg(\forall R.C) &\equiv (\exists R.\neg C) \end{aligned}$$

The soundness and completeness of the algorithm will be proved by showing that it creates a *tableau* for D:

Definition 2 If D is an $\mathcal{ALCI}_{R^+}$-concept in NNF and $\mathbf{R}_D$ is the set of roles occurring in D, together with their inverses, a tableau T for D is defined to be a triple $(\mathbf{S}, \mathcal{L}, \mathcal{E})$ such that: $\mathbf{S}$ is a set of individuals, $\mathcal{L} : \mathbf{S} \to 2^{sub(D)}$ maps each individual to a set of concepts which is a subset of $sub(D)$, $\mathcal{E} : \mathbf{R}_D \to 2^{\mathbf{S}\times\mathbf{S}}$ maps each role in $\mathbf{R}_D$ to a set of pairs of individuals, and there is some individual $s \in \mathbf{S}$ such that $D \in \mathcal{L}(s)$. For all $s \in \mathbf{S}$, $C, C_1, C_2 \in sub(D)$, and $R \in \mathbf{R}_D$, it holds that:

1. if $C \in \mathcal{L}(s)$, then $\neg C \notin \mathcal{L}(s)$,
2. if $C_1 \sqcap C_2 \in \mathcal{L}(s)$, then $C_1 \in \mathcal{L}(s)$ and $C_2 \in \mathcal{L}(s)$,
3. if $C_1 \sqcup C_2 \in \mathcal{L}(s)$, then $C_1 \in \mathcal{L}(s)$ or $C_2 \in \mathcal{L}(s)$,
4. if $\forall R.C \in \mathcal{L}(s)$ and $\langle s, t\rangle \in \mathcal{E}(R)$, then $C \in \mathcal{L}(t)$,
5. if $\exists R.C \in \mathcal{L}(s)$, then there is some $t \in \mathbf{S}$ such that $\langle s, t\rangle \in \mathcal{E}(R)$ and $C \in \mathcal{L}(t)$,
6. if $\forall R.C \in \mathcal{L}(s)$, $\langle s, t\rangle \in \mathcal{E}(R)$ and $\mathsf{Trans}(R)$, then $\forall R.C \in \mathcal{L}(t)$, and

7. $\langle x, y\rangle \in \mathcal{E}(R)$ iff $\langle y, x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$.

Lemma 1 *An $\mathcal{ALCI}_{R^+}$-concept D is satisfiable iff there exists a tableau for D.*

Proof: For the *if* direction, if $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ is a tableau for D with $D \in \mathcal{L}(s_0)$, a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of D can be defined as:

$$\begin{array}{rcl} \Delta^{\mathcal{I}} & = & \mathbf{S} \\ \mathsf{CN}^{\mathcal{I}} & = & \{s \mid \mathsf{CN} \in \mathcal{L}(s)\} \text{ for all concept names } \mathsf{CN} \text{ in } sub(D) \\ R^{\mathcal{I}} & = & \begin{cases} \mathcal{E}(R)^+ & \text{if } \mathsf{Trans}(R) \\ \mathcal{E}(R) & \text{otherwise} \end{cases} \end{array}$$

where $\mathcal{E}(R)^+$ denotes the transitive closure of $\mathcal{E}(R)$. $D^{\mathcal{I}} \neq \emptyset$ because $s_0 \in D^{\mathcal{I}}$. Transitive roles are obviously interpreted as transitive relations. By induction on the structure of concepts, we show that, if $E \in \mathcal{L}(s)$, then $s \in E^{\mathcal{I}}$. Let $E \in \mathcal{L}(s)$ with $E \in sub(D)$.

1. If E is a concept name, then $s \in E^{\mathcal{I}}$ by definition.

2. If $E = \neg C$, then $C \notin \mathcal{L}(s)$ (due to property 1 in Definition 2), so $s \in \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} = E^{\mathcal{I}}$.

3. If $E = (C_1 \sqcap C_2)$, then $C_1 \in \mathcal{L}(s)$ and $C_2 \in \mathcal{L}(s)$, so by induction $s \in C_1^{\mathcal{I}}$ and $C_2^{\mathcal{I}}$. Hence $s \in (C_1 \sqcap C_2)^{\mathcal{I}}$.

4. If $E = (C_1 \sqcup C_2)$, then $C_1 \in \mathcal{L}(s)$ or $C_2 \in \mathcal{L}(s)$, so by induction $s \in C_1^{\mathcal{I}}$ or $s \in C_2^{\mathcal{I}}$. Hence $s \in (C_1 \sqcup C_2)^{\mathcal{I}}$.

5. If $E = (\exists S.C)$, then there is some $t \in \mathbf{S}$ such that $\langle s, t\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)$. By definition, $\langle s, t\rangle \in S^{\mathcal{I}}$ and by induction $t \in C^{\mathcal{I}}$. Hence $S \in (\exists S.C)^{\mathcal{I}}$.

6. If $E = (\forall S.C)$ and $\langle s, t\rangle \in S^{\mathcal{I}}$, then either

   (a) $\langle s, t\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)$, or

   (b) $\langle s, t\rangle \notin \mathcal{E}(S)$, then there exists a path of length $n \geq 1$ such that $\langle s, s_1\rangle, \langle s_1, s_2\rangle, \ldots, \langle s_n, t\rangle \in \mathcal{E}(S)$. Due to property 6 in Definition 2, $\forall S.C \in \mathcal{L}(s_i)$ for all $1 \leq i \leq n$, and we have $C \in \mathcal{L}(t)$.

   In both cases, we have by induction $t \in C^{\mathcal{I}}$, hence $s \in (\forall S.C)^{\mathcal{I}}$.

For the converse, if $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of D, then a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ for D can be defined as:

$$\begin{aligned} \mathbf{S} &= \Delta^{\mathcal{I}} \\ \mathcal{E}(R) &= R^{\mathcal{I}} \\ \mathcal{L}(s) &= \{C \in sub(D) \mid s \in C^{\mathcal{I}}\} \end{aligned}$$

It only remains to demonstrate that T is a tableau for D:

1. T satisfies properties 1–5 in Definition 2 as a direct consequence of the semantics of the $\neg C$, $C_1 \sqcap C_2$, $C_1 \sqcup C_2$, $\forall R.C$ and $\exists R.C$ concept expressions.

2. If $d \in (\forall R.C)^{\mathcal{I}}$, $\langle d, e\rangle \in R^{\mathcal{I}}$ and $\mathsf{Trans}(R)$, then $e \in (\forall R.C)^{\mathcal{I}}$ unless there is some f such that $\langle e, f\rangle \in R^{\mathcal{I}}$ and $f \notin C^{\mathcal{I}}$. However, if $\langle d, e\rangle \in R^{\mathcal{I}}$, $\langle e, f\rangle \in R^{\mathcal{I}}$ and $R \in \mathbf{R}_+$, then $\langle d, f\rangle \in R^{\mathcal{I}}$ and $d \notin (\forall R.C)^{\mathcal{I}}$. T therefore satisfies property 6 in Definition 2.

3. T satisfies property 7 in Definition 2 as a direct consequence of the semantics of inverse relations. ∎

2.3 Constructing an $\mathcal{ALCI}_{R^+}$ Tableau

From Lemma 1, an algorithm which constructs a tableau for an $\mathcal{ALCI}_{R^+}$-concept D can be used as a decision procedure for the satisfiability of D. Such an algorithm will now be described in detail.

The tableaux algorithm works on *completion trees*. This is a tree where each node x of the tree is labelled with a set $\mathcal{L}(x) \subseteq sub(D)$ and each edge $\langle x, y\rangle$ is labelled $\mathcal{L}(\langle x, y\rangle) = R$ for some (possibly inverse) role R occurring in $sub(D)$. Edges are added when expanding $\exists R.C$ and $\exists R^-.C$ terms; they correspond to relationships between pairs of individuals and are always directed from the root node to the leaf nodes. The algorithm expands the tree either by extending $\mathcal{L}(x)$ for some node x or by adding new leaf nodes.

For a node x, $\mathcal{L}(x)$ is said to contain a *clash* if, for some concept C, $\{C, \neg C\} \subseteq \mathcal{L}(x)$.

If nodes x and y are connected by an edge $\langle x, y\rangle$, then y is called a *successor* of x and x is called a *predecessor* of y; *ancestor* is the transitive closure of *predecessor*.

$\sqcap$-rule:	if	1. $C_1 \sqcap C_2 \in \mathcal{L}(x)$, x is not indirectly blocked, and
		2. $\{C_1, C_2\} \not\subseteq \mathcal{L}(x)$
	then	$\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{C_1, C_2\}$
$\sqcup$-rule:	if	1. $C_1 \sqcup C_2 \in \mathcal{L}(x)$, x is not indirectly blocked, and
		2. $\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$
	then	$\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{C\}$ for some $C \in \{C_1, C_2\}$
$\exists$-rule:	if	1. $\exists S.C \in \mathcal{L}(x)$, x is not blocked, and
		2. x has no S-neighbour y with $C \in \mathcal{L}(y)$:
	then	create a new node y with $\mathcal{L}(\langle x, y\rangle) = S$ and $\mathcal{L}(y) = \{C\}$
$\forall$-rule:	if	1. $\forall S.C \in \mathcal{L}(x)$, x is not indirectly blocked, and
		2. there is an S-neighbour y of x with $C \notin \mathcal{L}(y)$:
	then	$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{C\}$
$\forall_+$-rule:	if	1. $\forall S.C \in \mathcal{L}(x)$, $\mathsf{Trans}(S)$, x is not indirectly blocked, and
		2. there is an S-neighbour y of x with $\forall S.C \notin \mathcal{L}(y)$
	then	$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\forall S.C\}$

Figure 1: Tableaux expansion rules for $\mathcal{ALCI}_{R^+}$

A node y is called an *R-neighbour* of a node x if either y is a successor of x and $\mathcal{L}(\langle x, y\rangle) = R$ or y is a predecessor of x and $\mathcal{L}(\langle y, x\rangle) = \mathsf{Inv}(R)$.

A node x is *blocked* if for some ancestor y, y is blocked or $\mathcal{L}(x) = \mathcal{L}(y)$. A blocked node x is *indirectly blocked* if its predecessor is blocked, otherwise it is *directly blocked*. If x is directly blocked it has a unique ancestor y such that $\mathcal{L}(x) = \mathcal{L}(y)$: if there existed another ancestor z such that $\mathcal{L}(x) = \mathcal{L}(z)$ then either y or z must be blocked. If x is directly blocked, and y is the unique ancestor such that $\mathcal{L}(x) = \mathcal{L}(y)$, we will say that y *blocks* x.

The algorithm initialises a tree $\mathbf{T}$ to contain a single node x_0, called the *root* node, with $\mathcal{L}(x_0) = \{D\}$, where D is the concept to be tested for satisfiability. $\mathbf{T}$ is then expanded by repeatedly applying the rules from Figure 1.

The completion tree is complete when for some node x, $\mathcal{L}(x)$ contains a clash, or when none of the rules is applicable. If, for an input concept D,

the expansion rules can be applied in such a way that they yield a complete, clash-free completion tree, then the algorithm returns "D is *satisfiable*", and "D is *unsatisfiable*" otherwise.

2.4 Soundness and Completeness

The soundness and completeness of the algorithm will be demonstrated by proving that, for an $\mathcal{ALCI}_{R^+}$-concept D, it always terminates and that it returns *satisfiable* if and only if D is satisfiable.

Lemma 2 *For each $\mathcal{ALCI}_{R^+}$-concept D, the tableaux algorithm terminates.*

Proof: Let $m = |sub(D)|$. Obviously, m is linear in the length of D. Termination is a consequence of the following properties of the expansion rules:

1. The expansion rules never remove nodes from the tree or concepts from node labels.

2. Successors are only generated for existential value restrictions (concepts of the form $\exists R.C$), and for any node each of these restrictions triggers the generation of at most one successor. Since $sub(D)$ cannot contain more than m existential value restrictions, the out-degree of the tree is bounded by m.

3. Nodes are labelled with nonempty subsets of $sub(D)$. If a path p is of length at least 2^m, then there are 2 nodes x, y on p, with $\mathcal{L}(x) = \mathcal{L}(y)$, and blocking occurs. Since a path on which nodes are blocked cannot become longer, paths are of length at most 2^m. ∎

Together with Lemma 1, the following lemma implies soundness of the tableaux algorithm.

Lemma 3 *If the expansion rules can be applied to an $\mathcal{ALCI}_{R^+}$-concept D such that they yield a complete and clash-free completion tree, then D has a tableau.*

Proof: Let $\mathbf{T}$ be the complete and clash-free tree constructed by the tableaux algorithm for D. A tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ can be defined with:

$$\mathbf{S} = \{x \mid x \text{ is a node in } \mathbf{T} \text{ and } x \text{ is not blocked}\},$$

$$\mathcal{E}(R) = \{\langle x, y\rangle \in \mathbf{S} \times \mathbf{S} \mid \begin{array}{ll} 1. & y \text{ is an } R\text{-neighbour of } x \quad \textit{or} \\ 2. & \mathcal{L}(\langle x, z\rangle) = R \text{ and } y \text{ blocks } z \quad \textit{or} \\ 3. & \mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R) \text{ and } x \text{ blocks } z\}, \end{array}$$

and it can be shown that T is a tableau for D:

1. $D \in \mathcal{L}(x_0)$ for the root x_0 of $\mathbf{T}$, and as x_0 has no predecessors it cannot be blocked. Hence $D \in \mathcal{L}(s)$ for some $s \in \mathbf{S}$.

2. Property 1 of Definition 2 is satisfied because $\mathbf{T}$ is clash free.

3. Properties 2 and 3 of Definition 2 are satisfied because neither the $\sqcap$-rule nor the $\sqcup$-rule apply to any $x \in \mathbf{S}$.

4. Property 4 in Definition 2 is satisfied because for all $x \in \mathbf{S}$, if $\forall R.C \in \mathcal{L}(x)$ and $\langle x, y\rangle \in \mathcal{E}(R)$ then either:

   (a) x is an R-neighbour of y,

   (b) $\mathcal{L}(\langle x, z\rangle) = R$, y blocks z, from the $\forall$-rule $C \in \mathcal{L}(z)$, $\mathcal{L}(y) = \mathcal{L}(z)$, or

   (c) $\mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R)$, x blocks z, $\mathcal{L}(x) = \mathcal{L}(z)$, so from the $\forall$-rule $C \in \mathcal{L}(y)$.

   In all 3 cases, the $\forall$-rule ensures that $C \in \mathcal{L}(y)$.

5. Property 5 in Definition 2 is satisfied because for all $x \in \mathbf{S}$, if $\exists R.C \in \mathcal{L}(x)$ then the $\exists$-rule ensures that there is either:

   (a) a predecessor y such that $\mathcal{L}(\langle y, x\rangle) = \mathsf{Inv}(R)$ and $C \in \mathcal{L}(y)$. Because y is a predecessor of x it cannot be blocked, so $y \in \mathbf{S}$ and $\langle y, x\rangle \in \mathcal{E}(R)$.

   (b) a successor y such that $\mathcal{L}(\langle x, y\rangle) = R$ and $C \in \mathcal{L}(y)$. If y is not blocked, then $y \in \mathbf{S}$ and $\langle x, y\rangle \in \mathcal{E}(R)$. Otherwise, y is blocked by some z with $\mathcal{L}(z) = \mathcal{L}(y)$. Hence $C \in \mathcal{L}(z)$, $z \in \mathbf{S}$ and $\langle x, z\rangle \in \mathcal{E}(R)$.

6. Property 6 in Definition 2 is satisfied because for all $x \in \mathbf{S}$, if $\forall R.C \in \mathcal{L}(x)$, $\langle x, y\rangle \in \mathcal{E}(R)$ and $\mathsf{Trans}(R)$ then either:

   (a) x is an R-neighbour of y,

(b) $\mathcal{L}(\langle x, z\rangle) = R$, y blocks z, and $\mathcal{L}(y) = \mathcal{L}(z)$, or

(c) $\mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R)$, x blocks z, hence $\mathcal{L}(x) = \mathcal{L}(z)$ and $\forall R.C \in \mathcal{L}(z)$.

In all 3 cases, the $\forall_+$-rule ensures that $\forall R.C \in \mathcal{L}(y)$.

7. Property 7 in Definition 2 is satisfied because for each $\langle x, y\rangle \in \mathcal{E}(R)$, either:

   (a) x is an R-neighbour of y, so y is an $\mathsf{Inv}(R)$-neighbour of x and $\langle y, x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$.

   (b) $\mathcal{L}(\langle x, z\rangle) = R$ and y blocks z, so $\mathcal{L}(\langle x, z\rangle) = \mathsf{Inv}(\mathsf{Inv}(R))$ and $\langle y, x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$.

   (c) $\mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R)$ and x blocks z, so $\langle y, x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$.

■

Lemma 4 *If D has a tableau, then the expansion rules can be applied in such a way that the tableaux algorithm yields a complete and clash-free completion tree for D.*

Proof: Let $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ be a tableau for D. Using T, we trigger the application of the expansion rules such that they yield a completion tree $\mathbf{T}$ that is both complete and clash-free. We start with $\mathbf{T}$ consisting of a single node x_0, the root, with $\mathcal{L}(x_0) = \{D\}$.

T is a tableau, hence there is some $s_0 \in \mathbf{S}$ with $D \in \mathcal{L}(s_0)$. When applying the expansion rules to $\mathbf{T}$, the application of the non-deterministic $\sqcup$-rule is driven by the labelling in the tableau T. To this purpose, we define a mapping π which maps the nodes of $\mathbf{T}$ to elements of $\mathbf{S}$, and we steer the application of the $\sqcup$-rule such that $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ holds for all nodes x of the completion tree.

More precisely, we define π inductively as follows:

- $\pi(x_0) = s_0$.
- If $\pi(x_i) = s_i$ is already defined, and a successor y of x_i was generated for $\exists R.C \in \mathcal{L}(x_i)$, then $\pi(y) = t$ for some $t \in \mathbf{S}$ with $C \in \mathcal{L}(t)$ and $\langle s_i, t\rangle \in \mathcal{E}(R)$.

To make sure that we have $\mathcal{L}(x_i) \subseteq \mathcal{L}(\pi(x_i))$, we use the $\sqcup'$-rule instead of the $\sqcup$-rule, where

$\sqcup'$-rule: if 1. $C_1 \sqcup C_2 \in \mathcal{L}(x)$, x is not indirectly blocked, and
2. $\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$
then $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{D\}$ for some $D \in \{C_1, C_2\} \cap \mathcal{L}(\pi(x))$,

The expansion rules given in Figure 1 with the $\sqcup$-rule replaced by the $\sqcup'$-rule are called *modified* expansion rules in the following.

It is easy to see that, if a tree $\mathbf{T}$ was generated using the modified expansion rules, then the expansion rules can be applied in such a way that they yield $\mathbf{T}$. Hence Lemma 3 and Lemma 2 still apply, and thus using the $\sqcup'$-rule instead of the $\sqcup$-rule preserves soundness and termination.

We will now show by induction that, if $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ holds for all nodes x in $\mathbf{T}$, then the application of an expansion rule preserves this subset-relation. To start with, we clearly have $\{D\} = \mathcal{L}(x_0) \subseteq \mathcal{L}(s_0)$.

If the $\sqcap$-rule can be applied to x in $\mathbf{T}$ with $C = C_1 \sqcap C_2 \in \mathcal{L}(x)$, then C_1, C_2 are added to $\mathcal{L}(x)$. Since T is a tableau, $\{C_1, C_2\} \subseteq \mathcal{L}(\pi(x))$, and hence the $\sqcap$-rule preserves the subset-relation between $\mathcal{L}(x)$ and $\mathcal{L}(\pi(x))$.

If the $\sqcup'$-rule can be applied to x in $\mathbf{T}$ with $C = C_1 \sqcup C_2 \in \mathcal{L}(x)$, then $D \in \{C_1, C_2\}$ is in $\mathcal{L}(\pi(x))$, and D is added to $\mathcal{L}(x)$ by the $\sqcup'$-rule. Hence the $\sqcup'$-rule preserves the subset-relation between $\mathcal{L}(x)$ and $\mathcal{L}(\pi(x))$.

If the $\exists$-rule can be applied to x in $\mathbf{T}$ with $C = \exists R.C_1 \in \mathcal{L}(x)$, then $C \in \mathcal{L}(\pi(x))$ and there is some $t \in \mathbf{S}$ with $\langle \pi(x), t \rangle \in \mathcal{E}(R)$ and $C_1 \in \mathcal{L}(t)$. The $\exists$-rule creates a new successor y of x for which $\pi(y) = t$ for some t with $C_1 \in \mathcal{L}(t)$. Hence we have $\mathcal{L}(y) = \{C_1\} \subseteq \mathcal{L}(\pi(y))$.

If the $\forall$-rule can be applied to x in $\mathbf{T}$ with $C = \forall R.C_1 \in \mathcal{L}(x)$ and y is an R-neighbour of x, then $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(R)$, and thus $C_1 \in \mathcal{L}(\pi(y))$. The $\forall$-rule adds C_1 to $\mathcal{L}(y)$ and thus preserves the subset-relation between $\mathcal{L}(x)$ and $\mathcal{L}(\pi(x))$.

If the $\forall_+$-rule can be applied to x in $\mathbf{T}$ with $C = \forall R.C_1 \in \mathcal{L}(x)$, $\mathsf{Trans}(R)$ and y being an R-neighbour of x, then $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(R)$, and thus $\forall R.C_1 \in \mathcal{L}(\pi(y))$. The $\forall_+$-rule adds C_1 to $\mathcal{L}(y)$ and thus preserves the subset-relation between $\mathcal{L}(y)$ and $\mathcal{L}(\pi(y))$.

Summing up, the tableau-construction triggered by T terminates with a complete tree, and since $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ holds for all nodes x in $\mathbf{T}$, $\mathbf{T}$ is clash-free due to Property 1 of Definition 2. ∎

Theorem 1 *The tableaux algorithm is a decision procedure for the satisfiability and subsumption of $\mathcal{ALCI}_{R^+}$-concepts.*

Theorem 1 is an immediate consequence of the Lemmata 1, 2, 3, and 4. Moreover, since $\mathcal{ALCI}_{R^+}$ is closed under negation, subsumption $C \sqsubseteq D$ can be reduced to unsatisfiability of $C \sqcap \neg D$.

3 Extending $\mathcal{ALCI}_{R^+}$ by Role Hierarchies

We will now extend the tableaux algorithm presented in Section 2.3 to deal with *role hierarchies* in a similar way to the algorithm for $\mathcal{ALCH}_{R^+}$ presented in [Horrocks & Gough,1997]. $\mathcal{ALCHI}_{R^+}$ extends $\mathcal{ALCI}_{R^+}$ by allowing, additionally, for inclusion axioms on roles. These axioms can involve transitive as well as non-transitive roles, and inverse roles as well as role names. For example, to express that a role R is symmetric, we add the two axioms $R \sqsubseteq R^-$ and $R^- \sqsubseteq R$.

Definition 3 A *role inclusion axiom* is of the form

$$R \sqsubseteq S,$$

for two (possibly inverse) roles R and S. For a set of role inclusion axioms $\mathcal{R}$, $\mathcal{R}^+ := (\mathcal{R} \cup \{\mathsf{Inv}(R) \sqsubseteq \mathsf{Inv}(S) \mid R \sqsubseteq S \in \mathcal{R}\}, \boxed{*}{\sqsubseteq})$ is called a *role hierarchy*, where $\boxed{*}{\sqsubseteq}$ is the transitive-reflexive closure of $\sqsubseteq$ over $\mathcal{R} \cup \{\mathsf{Inv}(R) \sqsubseteq \mathsf{Inv}(S) \mid R \sqsubseteq S \in \mathcal{R}\}$.

Definition 4 $\mathcal{ALCHI}_{R^+}$ is the extension of $\mathcal{ALCI}_{R^+}$ obtained by allowing, additionally, for a role hierarchy $\mathcal{R}^+$.

As well as being correct for $\mathcal{ALCI}_{R^+}$ concepts, an $\mathcal{ALCHI}_{R^+}$ interpretation has to satisfy the additional condition,

$$\langle x, y\rangle \in R^{\mathcal{I}} \text{ implies } \langle x, y\rangle \in S^{\mathcal{I}} \text{ for all roles} R, S \text{ with } R \boxed{*}{\sqsubseteq} S.$$

The tableaux algorithm given in the preceding section can easily be modified to decide satisfiability of $\mathcal{ALCHI}_{R^+}$-concepts by extending the definitions of both R-neighbours and the $\forall_+$-rule to include the notion of role hierarchies. To prove the soundness and correctness of the extended algorithm, the definition of a tableau is also extended.

Definition 5 As well as satisfying Definition 2 (i.e. being a valid $\mathcal{ALCI}_{R^+}$ tableau), a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ for an $\mathcal{ALCHI}_{R^+}$-concept D must also satisfy:

$\forall'_+$-rule:	if	1.	$\forall S.C \in \mathcal{L}(x)$, x is not indirectly blocked, and
		2.	there is some R with $\mathsf{Trans}(R)$ and $R \sqsubseteq^* S$,
		3.	there is an R-neighbour y of x with $\forall R.C \notin \mathcal{L}(y)$
	then		$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\forall R.C\}$

Figure 2: The new $\forall_+$-rule for $\mathcal{ALCHI}_{R^+}$.

6′. if $\forall S.C \in \mathcal{L}(s)$ and $\langle s,t\rangle \in \mathcal{E}(R)$ for some $R \sqsubseteq^* S$ with $\mathsf{Trans}(R)$, then $\forall R.C \in \mathcal{L}(t)$,

8. if $\langle x,y\rangle \in \mathcal{E}(R)$ and $R \sqsubseteq^* S$, then $\langle x,y\rangle \in \mathcal{E}(S)$,

where property 6′ extends and supersedes property 6 from Definition 2.

For the $\mathcal{ALCHI}_{R^+}$ algorithm, the $\forall_+$-rule is replaced with the $\forall'_+$-rule (see Figure 2), and the definition of R-neighbours extended as follows:

Definition 6 Given a completion tree, a node y is called an *R-neighbour* of a node x if either y is a successor of x and $\mathcal{L}(\langle x,y\rangle) = S$ or y is a predecessor of x and $\mathcal{L}(\langle y,x\rangle) = \mathsf{Inv}(S)$ for some S with $S \sqsubseteq^* R$.

In the following, the tableaux algorithm resulting from these modifications will be called the *modified tableaux algorithm*.

To prove that the modified tableaux algorithm is indeed a decision procedure for the satisfiability of $\mathcal{ALCHI}_{R^+}$-concepts, all 4 technical lemmata used in Section 2 to prove this fact for the $\mathcal{ALCI}_{R^+}$ tableaux algorithm have to be re-proven for $\mathcal{ALCHI}_{R^+}$. In the following, we will restrict our attention to cases that differ from those already considered for $\mathcal{ALCI}_{R^+}$.

Lemma 5 *An $\mathcal{ALCHI}_{R^+}$-concept D is satisfiable iff there exists a tableau for D.*

Proof: For the *if* direction, the construction of a model of D from a tableau for D is similar to the one presented in the proof of Lemma 1. If $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ is a tableau for D with $D \in \mathcal{L}(s_0)$, a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of D can be defined as follows:

$$\begin{aligned} \Delta^{\mathcal{I}} &= \mathbf{S} \\ \mathsf{CN}^{\mathcal{I}} &= \{s \mid \mathsf{CN} \in \mathcal{L}(s)\} \text{ for all concept names } \mathsf{CN} \text{ in } sub(D) \end{aligned}$$

$$R^{\mathcal{I}} = \begin{cases} \mathcal{E}(R)^{+} & \text{if } \mathsf{Trans}(R) \\ \mathcal{E}(R) \cup \bigcup_{P \sqsubseteq^{*} R, P \neq R} P^{\mathcal{I}} & \text{otherwise} \end{cases}$$

The interpretation of non-transitive roles is recursive in order to correctly interpret those non-transitive roles that have a transitive sub-role. From the definition of $R^{\mathcal{I}}$ and property 8 of a tableau it follows that if $\langle x, y\rangle \in S^{\mathcal{I}}$, then either $\langle x, y\rangle \in \mathcal{E}(S)$ or there exists a path $\langle s, s_1\rangle, \langle s_1, s_2\rangle, \ldots, \langle s_n, t\rangle \in \mathcal{E}(R)$ for some R with $\mathsf{Trans}(R)$ and $R \sqsubseteq^{*} S$.

Property 8 of a tableau ensures that $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ holds for all roles with $R \sqsubseteq^{*} S$, including those cases where R is a transitive role. Again, it can be shown by induction on the structure of concepts that $\mathcal{I}$ is a correct interpretation. We restrict our attention to the only case that is different from the ones in the proof of Lemma 1. Let $E \in sub(D)$ with $E \in \mathcal{L}(s)$.

6′. If $E = (\forall S.C)$ and $\langle s, t\rangle \in S^{\mathcal{I}}$, then either

(a) $\langle s, t\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)$, or

(b) $\langle s, t\rangle \notin \mathcal{E}(S)$, then there exists a path of length $n \geq 1$ such that $\langle s, s_1\rangle, \langle s_1, s_2\rangle, \ldots, \langle s_n, t\rangle \in \mathcal{E}(R)$ for some R with $\mathsf{Trans}(R)$ and $R \sqsubseteq^{*} S$. Due to Property 6', $\forall R.C \in \mathcal{L}(s_i)$ for all $1 \leq i \leq n$, and we have $C \in \mathcal{L}(t)$.

In both cases, we have $t \in C^{\mathcal{I}}$.

For the converse, if $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of D, then a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ for D can be defined as:

$$\begin{aligned} \mathbf{S} &= \Delta^{\mathcal{I}} \\ \mathcal{E}(R) &= R^{\mathcal{I}} \\ \mathcal{L}(s) &= \{C \in sub(D) \mid s \in C^{\mathcal{I}}\} \end{aligned}$$

It remains to demonstrate that T is a tableau for D:

1. T satisfies properties 1–5 in Definition 2 as a direct consequence of the semantics of $\mathcal{ALCHI}_{R^+}$-concepts.

2. If $d \in (\forall S.C)^{\mathcal{I}}$ and $\langle d, e\rangle \in R^{\mathcal{I}}$ for R with $\mathsf{Trans}(R)$ and $R \sqsubseteq^{*} S$, then $e \in (\forall R.C)^{\mathcal{I}}$ unless there is some f such that $\langle e, f\rangle \in R^{\mathcal{I}}$ and $f \notin C^{\mathcal{I}}$. In this case, if $\langle d, e\rangle \in R^{\mathcal{I}}$, $\langle e, f\rangle \in R^{\mathcal{I}}$ and $\mathsf{Trans}(R)$, then $\langle d, f\rangle \in R^{\mathcal{I}}$. Hence $\langle d, f\rangle \in S^{\mathcal{I}}$ and $d \notin (\forall S.C)^{\mathcal{I}}$—in contradiction of the assumption. T therefore satisfies Property 6′ in Definition 5.

3. Since $\mathcal{I}$ is a model of D, $\langle x, y\rangle \in R^{\mathcal{I}}$ implies $\langle x, y\rangle \in S^{\mathcal{I}}$ for all roles R, S with $R \sqsubseteq^{*} S$. Hence T satisfies Property 8 in Definition 5. ∎

Lemma 6 *For each $\mathcal{ALCHI}_{R^+}$-concept D, the modified tableaux algorithm terminates.*

The proof is identical to the one given for Lemma 2.

Lemma 7 *If the expansion rules can be applied to an $\mathcal{ALCHI}_{R^+}$-concept D such that they yield a complete and clash-free completion tree, then D has a tableau.*

Proof: The definition of a tableau from a complete and clash-free completion tree, as presented in the proof of Lemma 3, has to be slightly modified. A tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ is now defined with:

$$
\begin{array}{rcl}
\mathbf{S} & = & \{x \mid \ x \text{ is a node in } \mathbf{T} \text{ and } x \text{ is not blocked}\} \\
\mathcal{E}(S) & = & \{\langle x, y\rangle \in \mathbf{S} \times \mathbf{S} \mid \ 1.\ y \text{ is an } S\text{-neighbour of } x \quad \text{or} \\
 & & \qquad\quad 2.\ \text{There exists a role } R \text{ with } R \sqsubseteq^{*} S \text{ and} \\
 & & \qquad\qquad a.\ \mathcal{L}(\langle x, z\rangle) = R \text{ and } y \text{ blocks } z \quad \text{or} \\
 & & \qquad\qquad b.\ \mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R) \text{ and } x \text{ blocks } z\}
\end{array}
$$

and, again, it can be shown that T is a tableau for D:

1. Since the expansion rules were started with $\mathcal{L}(x_0) = \{D\}$, $D \in \mathcal{L}(x_0)$ for some $x_0 \in \mathbf{S}$.

2. Properties 1-3 are identical to the proof of Lemma 3.

3. Property 4 in Definition 2 is satisfied because for all $x \in \mathbf{S}$, if $\forall S.C \in \mathcal{L}(x)$ and $\langle x, y\rangle \in \mathcal{E}(S)$ then either:

   (a) x is an S-neighbour of y,

   (b) for some role with $R \sqsubseteq^{*} S$, either

      i. $\mathcal{L}(\langle x, z\rangle) = R$, y blocks z, hence from the $\forall$-rule $C \in \mathcal{L}(z)$, and $\mathcal{L}(y) = \mathcal{L}(z)$, or

      ii. $\mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R)$, x blocks z, hence $\mathcal{L}(x) = \mathcal{L}(z)$ and therefor $\forall S.C \in \mathcal{L}(z)$.

   In all three cases, the $\forall$-rule ensures $C \in \mathcal{L}(y)$.

4. Property 5 in Definition 2 is satisfied for the same reasons as in the proof of Lemma 3

5. Property 6' in Definition 5 is satisfied because for all $x \in \mathbf{S}$, if $\forall S.C \in \mathcal{L}(x)$, $\langle x, y\rangle \in \mathcal{E}(R)$ for some R with $\mathsf{Trans}(R)$ and $R \boxed{*} S$, then either:

   (a) y is an R-neighbor of x, or

   (b) there is some role R' with $R' \sqsubseteq R$ and

      i. $\mathcal{L}(\langle x, z\rangle) = R'$, y blocks z and $\mathcal{L}(y) = \mathcal{L}(z)$, or

      ii. $\mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R)$, x blocks z and $\mathcal{L}(x) = \mathcal{L}(z)$, hence $\forall S.C \in \mathcal{L}(z)$.

   In all three cases, $\forall R.C \in \mathcal{L}(y)$ follows from the $\forall'_+$-rule.

6. Property 8 in Definition 5 follows immediately from the definition of $\mathcal{E}$. ∎

Lemma 8 *If $\mathcal{ALCHI}_{R^+}$-concept D has a tableau, then the expansion rules can be applied in such a way that the tableaux algorithm yields a complete and clash-free completion tree for D.*

The proof of Lemma 8 is identical to the one presented for Lemma 4. Again, summing up, we have the following theorem.

Theorem 2 *The tableaux algorithm is a decision procedure for the satisfiability and subsumption of $\mathcal{ALCHI}_{R^+}$-concepts.*

3.1 General Concept Inclusion Axioms

In [Baader,1991; Schild,1991; Baader *et al.*,1993], the *internalisation* of terminological axioms is introduced. This technique is used to reduce reasoning with respect to a (possibly cyclic) *terminology* to satisfiability of concepts. In [Horrocks & Gough,1997], we saw how role hierarchies can be used to reduce satisfiability and subsumption with respect to a terminology to concept satisfiability and subsumption. In the presence of inverse roles, this reduction must be slightly modified.

Definition 7 A *terminology* $\mathcal{T}$ is a finite set of general concept inclusion axioms,

$$\mathcal{T} = \{C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n\},$$

where C_i, D_i are arbitrary $\mathcal{ALCHI}_{R^+}$-concepts. An interpretation $\mathcal{I}$ is said to be a model of $\mathcal{T}$ iff $C_i^{\mathcal{I}} \subseteq D_i^{\mathcal{I}}$ holds for all $C_i \sqsubseteq D_i \in \mathcal{T}$. C is satisfiable with respect to $\mathcal{T}$ iff there is a model $\mathcal{I}$ of $\mathcal{T}$ with $C^{\mathcal{I}} \neq \emptyset$. Finally, D subsumes C with respect to $\mathcal{T}$ ($C \sqsubseteq_{\mathcal{T}} D$) iff for each model $\mathcal{I}$ of $\mathcal{T}$ we have $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.

The following lemma shows how general concept inclusion axioms can be *internalised* using a "universal" role U. This role U is a transitive super-role of all relevant roles and their respective inverses. Hence, for each interpretation $\mathcal{I}$, each individual t reachable via some role path from another individual s is an $U^{\mathcal{I}}$-successor of s. All general concept inclusion axioms $C_i \sqsubseteq D_i$ in $\mathcal{T}$ are propagated along all role paths using the value restriction $\forall U.\neg C \sqcup D$.

Lemma 9 *Let $\mathcal{T}$ be terminology and C, D be $\mathcal{ALCHI}_{R^+}$-concepts and let*

$$C_{\mathcal{T}} := \bigsqcap_{C_i \sqsubseteq D_i \in \mathcal{T}} \neg C_i \sqcup D_i.$$

Let U be a transitive role with $R \sqsubseteq U$, $\mathsf{Inv}(R) \sqsubseteq U$ for each role R that occurs in $\mathcal{T}, C,$ or D.

Then C is satisfiable with respect to $\mathcal{T}$ iff

$$C \sqcap C_{\mathcal{T}} \sqcap \forall U.C_{\mathcal{T}}$$

is satisfiable. D subsumes C with respect to $\mathcal{T}$ ($C \sqsubseteq_{\mathcal{T}} D$) iff

$$C \sqcap \neg D \sqcap C_{\mathcal{T}} \sqcap \forall U.C_{\mathcal{T}}$$

is unsatisfiable.

Remark: Instead of defining U as a transitive super-role of all roles and their respective inverses, one could have defined U as a transitive super-role of all roles and, additionally, a symmetric role by adding $U \sqsubseteq U^-$ and $U^- \sqsubseteq U$.

The proof of Lemma 9 is similar to the ones that can be found in [Schild,1991; Baader,1990]. One point to show is that, if an $\mathcal{ALCHI}_{R^+}$-concept C is satisfiable with respect to a terminology $\mathcal{T}$, then $C, \mathcal{T}$ have a *connected* model, namely one whose individuals are all related to each

other by some role path. This follows from the definition of the semantics of $\mathcal{ALCHI}_{R^+}$-concepts. The other point to proof is that, if y is reachable from x via a role path (possibly involving inverse roles), then $\langle x, y\rangle \in U^{\mathcal{I}}$, which is an easy consequence of the definition of U.

Decidability of satisfiability and subsumption with respect to a terminology is an immediate consequence of Lemma 9 and Theorem 2.

Theorem 3 *The modified tableaux algorithm is a decision procedure for satisfiability and subsumption of $\mathcal{ALCHI}_{R^+}$-concepts with respect to terminologies.*

References

[Baader *et al.*, 1993] F. Baader, H.-J. Bürckert, B. Nebel, W. Nutt, and G. Smolka. On the expressivity of feature logics with negation, functional uncertainty, and sort equations. *Journal of Logic, Language and Information*, 2:1–18, 1993.

[Baader, 1990] F. Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. Technical Report RR-90-13, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern, Germany, 1990. An abridged version appeared in *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence IJCAI-91*, pp. 446–451.

[Baader, 1991] F. Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, 1991.

[De Giacomo & Lenzerini, 1996] G. De Giacomo and M. Lenzerini. Tbox and Abox reasoning in expressive description logics. In *Proceedings of the Fifth International Conference on the Principles of Knowledge Representation and Reasoning (KR-96)*, pages 316–327. Morgan Kaufmann, Los Altos, 1996.

[De Giacomo & Massacci, 1998] G. De Giacomo and F. Massacci. Combining deduction and model checking into tableaux and algorithms for converse-pdl. *Information and Computation*, 1998. To appear.

[Horrocks & Gough, 1997] I. Horrocks and G. Gough. Description logics with transitive roles. In M.-C. Rousset, R. Brachmann, F. Donini, E. Franconi, I. Horrocks, and A. Levy, editors, *Proceedings of the International Workshop on Description Logics*, pages 25–28, Gif sur Yvette, France, 1997. Université Paris-Sud.

[Sattler, 1996] U. Sattler. A concept language extended with different kinds of transitive roles. In G. Görz and S. Hölldobler, editors, *20. Deutsche Jahrestagung für Künstliche Intelligenz*, volume 1137 of *Lecture Notes in Mathematics*. Springer-Verlag, 1996.

[Sattler, 1998] U. Sattler. *Terminological knowledge representation systems in a process engineering application.* PhD thesis, RWTH Aachen, 1998. To appear.

[Schild, 1991] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 466–471, Sydney, 1991.

[Schmidt-Schauß & Smolka, 1988] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with unions and complements. Technical Report SR-88-21, FB Informatik, Universität Kaiserslautern, Kaiserslautern, Germany, 1988.