TECHNISCHE UNIVERSITÄT DRESDEN

Dresden University of Technology
Institute for Theoretical Computer Science
Chair for Automata Theory

LTCS–Report

A Description Logic with Transitive and Converse Roles, Role Hierarchies and Qualifying Number Restrictions

Ian Horrocks and Ulrike Sattler and Stephan Tobies

LTCS-Report 99-08

Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
http://lat.inf.tu-dresden.de

Hans-Grundig-Str. 25
01062 Dresden
Germany

A Description Logic with Transitive and Converse Roles, Role Hierarchies and Qualifying Number Restrictions

Ian Horrocks and Ulrike Sattler and Stephan Tobies

Revised Version - March 11, 2004

Sections 1–3 have already been published as the LTCS-Report [HS98], Section 4 is an excerpt from [HS99]. The new material in this report extends these results and is presented in Sections 5 and 6.

Contents

1 Introduction

As widely argued [HG97; Sat96], 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 [HG97], the Description Logic (DL) $\mathcal{ALCH}_{R^+}$ is presented, which extends $\mathcal{ALC}$ with transitive roles and a role hierarchy. It is argued in [Sat98] 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 $\mathcal{SHI}$ 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 [DL96] 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 [HG97]), 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 [DM98], and this rule leads to an algorithm with very bad behaviour. We will present an algorithm for $\mathcal{SHI}$ without such a rule.

Furthermore, we enrich the language with functional restrictions and, finally, with qualifying number restrictions. We give sound and complete decision procedures for the resulting logics that are derived from the initial algorithm for $\mathcal{SHI}$.

The structure of this report is as follows: In Section 2, we introduce the DL $\mathcal{SI}$ and present a tableaux algorithm for satisfiability (and subsumption) of $\mathcal{SI}$-concepts—in another report [HST98] we prove that this algorithm can be refined to run in polynomial space. In Section 3 we add role hierarchies to $\mathcal{SI}$ and show how the algorithm can be modified to handle this extension appropriately. Please note that this logic, namely $\mathcal{SHI}$, allows for the internalisation of general concept inclusion axioms, one of the most general form of terminological axioms. In Section 4 we augment $\mathcal{SHI}$ with functional restrictions and, using the so-called pairwise-blocking technique, the algorithm can be adapted to this extension as well. Finally, in Section 5, we show that standard techniques for handling qualifying number restrictions [HB91; BBH96] together with the techniques described in previous sections can be used to decide satisfiability and subsumption for $\mathcal{SHIQ}$, namely $\mathcal{ALC}$ extended with transitive and inverse roles, role hierarchies, and qualifying number restrictions. Although Section 5 heavily depends on the previous sections, we have made it self-contained, i.e. it contains all necessary definitions and proofs from scratch, for a better readability. Building on the previous sections, Section 6 presents an algorithm that decides the satisfiability of $\mathcal{SHIQ}$-ABoxes.

2 A Tableaux Algorithm for $\mathcal{SI}$

In this section a tableaux algorithm for testing the satisfiability of $\mathcal{SI}$ concept expressions will be described and a proof of its soundness and completeness presented. The algorithm and proof are extensions of those described for $\mathcal{ALC}_{R^+}$ [Sat96].

2.1 Syntax and Semantics

$\mathcal{SI}$ is the Description Logic (DL) obtained by augmenting the well-known DL $\mathcal{ALC}$ [SS88] 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 $\mathcal{SI}$*-roles* is $\mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$. The set of $\mathcal{SI}$*-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 $\mathcal{SI}$-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

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

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 $\mathcal{SI}$ Tableau

Like other tableaux algorithms, the $\mathcal{SI}$ 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 $\mathcal{SI}$-concepts. When testing the satisfiability of an $\mathcal{SI}$-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 $\mathcal{SI}$-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 $\mathcal{SI}$-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 $\mathcal{SI}$-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:

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

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 $\mathcal{SI}$ Tableau

From Lemma 1, an algorithm which constructs a tableau for an $\mathcal{SI}$-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 $\mathcal{L}(x) \subseteq \mathit{sub}(D)$ and each edge $\langle x,y\rangle$ is labelled $\mathcal{L}(\langle x,y\rangle) = R$ for some (possibly inverse) role R occurring in $\mathit{sub}(D)$. 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*.

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 $\mathcal{L}(\langle y,x\rangle) = \mathsf{Inv}(R)$.

$\sqcap$-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 $\mathcal{SI}$

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 called *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 $\mathcal{SI}$-concept D, it always terminates and that it returns *satisfiable* if and only if D is satisfiable.

Lemma 2 *For each $\mathcal{SI}$-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 $\mathcal{SI}$-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:

$$\begin{array}{rcl} \mathbf{S} & = & \{x \mid x \text{ is a node in } \mathbf{T} \text{ and } x \text{ is not blocked}\}, \\ \mathcal{E}(R) & = & \{\langle x, y\rangle \in \mathbf{S} \times \mathbf{S} \mid \begin{array}{ll} 1. & y \text{ is an } R\text{-neighbour of } x \quad \textit{or} \\ 2. & \mathcal{L}(\langle x, z\rangle) = R \text{ and } y \text{ blocks } z \quad \textit{or} \\ 3. & \mathcal{L}(\langle y, z\rangle) = \mathsf{Inv}(R) \text{ and } x \text{ blocks } z\}, \end{array} \end{array}$$

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 $\mathcal{SI}$-concepts.*

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

3 Extending $\mathcal{SI}$ 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 [HG97]. $\mathcal{SHI}$ extends $\mathcal{SI}$ 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 $\mathcal{R}$, $\mathcal{R}^+ := (\mathcal{R} \cup \{\mathsf{Inv}(R) \sqsubseteq \mathsf{Inv}(S) \mid R \sqsubseteq S \in \mathcal{R}\}, \sqsubseteq^*)$ is called a *role hierarchy*, where $\sqsubseteq^*$ 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 $\mathcal{SHI}$ is the extension of $\mathcal{SI}$ obtained by allowing, additionally, for a role hierarchy $\mathcal{R}^+$.

As well as being correct for $\mathcal{SI}$ concepts, an $\mathcal{SHI}$ interpretation has to satisfy the additional condition,

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

The tableaux algorithm given in the preceding section can easily be modified to decide satisfiability of $\mathcal{SHI}$-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 $\mathcal{SI}$ tableau), a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ for an $\mathcal{SHI}$-concept D must also satisfy:

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

$\forall'_+$-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 $\mathcal{SHI}$.

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 $\mathcal{SHI}$-concepts, all 4 technical lemmata used in Section 2 to prove this fact for the $\mathcal{SI}$ tableaux algorithm have to be re-proven for $\mathcal{SHI}$. In the following, we will restrict our attention to cases that differ from those already considered for $\mathcal{SI}$.

Lemma 5 *An $\mathcal{SHI}$-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:

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

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 $\mathcal{SHI}$-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 $\mathcal{SHI}$-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 $\mathcal{SHI}$-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:

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

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 $R \sqsubseteq^{*} S$, 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 $\mathcal{SHI}$-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 $\mathcal{SHI}$-concepts.*

3.1 General Concept Inclusion Axioms

In [Baa90; Sch91; BBN+93], 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 [HG97], 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 $\mathcal{SHI}$-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 $\mathcal{SHI}$-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 [Sch91; Baa90]. One point to show is that, if an $\mathcal{SHI}$-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 $\mathcal{SHI}$-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 $\mathcal{SHI}$-concepts with respect to terminologies.*

4 Extending $\mathcal{SHI}$ by Functional Restrictions

In this section, we will present the extension of $\mathcal{SHI}$ with functional restrictions to give $\mathcal{SHIF}$. The most general way to do this is to allow, for (possibly inverse) roles R, concepts of the form $(\leqslant 1\ R)$. These concepts express local functionality, and can be used to express global functionality, by using the general concept inclusion axiom $\top \sqsubseteq (\leqslant 1\ R)$. As the logic supports general negation, it is also necessary to allow for negated functional restrictions $\neg(\leqslant 1\ R)$; in negation normal form these become restrictions of the form $(\geqslant 2\ R)$ [HNS90].

In $\mathcal{SHIF}$, the roles that can appear in functional restrictions are limited to *simple* roles, where a role is simple if it is neither transitive nor has transitive sub-roles. Without this limitation the extension of the $\mathcal{SHI}$ tableau construction algorithm would be more difficult due to the possibility of having to collapse a chain of successors into a single node. This would be necessary if, for example, $(\leqslant 1\ S)$ is added to the label of a node x where $R \in \mathbf{R}_+$, x already has a chain of R-successors, and $R \sqsubseteq^* S$.

Definition 8 $\mathcal{SHIF}$ is the extension of $\mathcal{SHI}$ obtained by allowing, additionally, for functional restrictions: for a simple role R, $(\leqslant 1\ R)$ is also an $\mathcal{SHIF}$-concept. A role R is a *simple* role iff $R \notin \mathbf{R}_+$ and, for any $S \sqsubseteq^* R$, S is also a simple role.

An $\mathcal{SHIF}$-interpretation is an $\mathcal{SHI}$-interpretation that satisfies, additionally,

$$(\leqslant 1\ R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{For all } y, z\text{: if } \langle x, y\rangle \in R^{\mathcal{I}} \text{ and } \langle x, z\rangle \in R^{\mathcal{I}}, \text{ then } y = z\},$$
$$(\geqslant 2\ R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{There exist } y, z\text{: } \langle x, y\rangle \in R^{\mathcal{I}}, \langle x, z\rangle \in R^{\mathcal{I}}, \text{ and } y \neq z\}.$$

Definition 9 If D is an $\mathcal{SHIF}$-concept in NNF, then a tableau T for D is defined like in Definition 5, with the additional properties:

9. if $(\leqslant 1\ R) \in \mathcal{L}(s)$ and $\langle s, t\rangle \in \mathcal{E}(R)$ and $\langle s, t'\rangle \in \mathcal{E}(R)$, then $t = t'$, and
10. if $(\geqslant 2\ R) \in \mathcal{L}(s)$, then there are some $t, t' \in \mathbf{S}$ such that $\langle s, t\rangle \in \mathcal{E}(R)$, $\langle s, t'\rangle \in \mathcal{E}(R)$, and $t \neq t'$.

Lemma 10 *An $\mathcal{SHIF}$-concept D is satisfiable iff there exists a tableau for D.*

As will be shown in the next section, $\mathcal{SHIF}$ no longer has the finite model property. In the algorithm presented here, this will be dealt with by generating (finite) completion trees and showing how they can be interpreted as infinite tableaux.

The proof is similar to the proof of Lemma 5, with the additional observations that

1. In the *if* direction, Properties 9 and 10 in Definition 9 ensure that functional restrictions are interpreted correctly. This depends on the fact that only simple roles can appear in functional restrictions, as for a simple role R, $R^{\mathcal{I}} = \mathcal{E}(R)$.

2. In the *only if* direction, the semantics of functional restrictions ensure that Properties 9 and 10 in Definition 9 are satisfied.

4.1 Pair-wise Blocking

Further extending the logic with functional restrictions (concepts of the form $(\leqslant 1\ R)$, meaning that an individual can be related to at most one other individual by the role R) and a role hierarchy (subsumption relationships between roles) introduces new problems associated with the fact that the logic no longer has the finite model property. This means that there are concepts that are satisfiable but for which there exists no finite model. An example of such a concept is

$$\neg C \sqcap \exists F^{-}.C \sqcap (\leqslant 1\ F) \sqcap \forall R^{-}.(\exists F^{-}.(C \sqcap (\leqslant 1\ F)))$$

where R is a transitive role and $F \sqsubseteq R$. Any model of this concept must contain an infinite sequence of individuals, each related to a single successor by an F^{-} role, and each satisfying $C \sqcap \exists F^{-}.C$, the $\exists F^{-}.C$ term being propagated along the sequence by the transitive super-role R. Attempting to terminate the sequence in a cycle causes the whole sequence to collapse into a single node due the functional restrictions $(\leqslant 1\ F)$, and this results in a contradiction as both C and $\neg C$ will be in the node's label.

In order to deal with infinite models—namely to have an algorithm that terminates correctly even if the input concept has only infinite models—a more sophisticated *pair-wise* blocking strategy is introduced, and soundness is proved by demonstrating that a (possibly blocked) tree always has a corresponding (possibly infinite) model.[1]

The new blocking strategy generates a tree where an infinite tableau is defined by recursively replacing the blocked node with a copy of the tree rooted at the blocking node. To be certain that this transplanted tree is still valid in its new location, blocks are established between pairs of nodes connected by the same role: a node y is blocked by a node x when their labels are equal, the labels of their predecessors y' and x' are equal, and the edges connecting x' to x

[1] This is not to say that it may not also have a finite model.

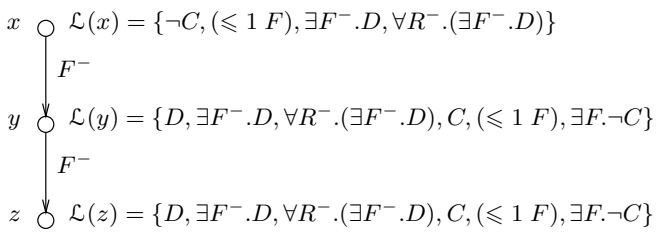

Figure 3: A tableau where pair-wise blocking is crucial

and y' to y are labelled with the same role names. Note the similarity between this condition and that imposed by the combination of the blocking condition and cut rule in 4*converse*-PDL [DM98].

Figure 3 shows how pair-wise blocking is crucial in order to ensure that the algorithm discovers the unsatisfiability of the concept

$$\neg C \sqcap (\leqslant 1\ F) \sqcap \exists F^-.D \sqcap \forall R^-.(\exists F^-.D),$$

where $F \sqsubseteq R$ and D represents the concept

$$C \sqcap (\leqslant 1\ F) \sqcap \exists F.\neg C.$$

Using dynamic blocking, z would be blocked by y. The resulting tree cannot represent a cyclical model in which y is related to itself by an F^- role as this would conflict with $(\leqslant 1\ F) \in \mathcal{L}(y)$. The tree must therefore represent the infinite model generated by recursively replacing each occurrence of z with a copy of the tree rooted at y. However, this is not a valid model as when z is substituted by a copy of y, $\exists F.\neg C \in \mathcal{L}(y)$, which was satisfied by $\neg C \in \mathcal{L}(x)$, is no longer satisfied in its new location.

When pair-wise blocking is used, z is no longer blocked by y as the labels of their predecessors (y and x respectively) are not equal, and the algorithm continues to expand $\mathcal{L}(z)$. The expansion of $\exists F.\neg C \in \mathcal{L}(z)$ calls for the existence of a node whose label includes $\neg C$ and that is connected to z by an F labelled edge. Because of $(\leqslant 1\ F) \in \mathcal{L}(z)$, this node must be y, and this results in a contradiction as both C and $\neg C$ will be in $\mathcal{L}(y)$.

4.2 Constructing an $\mathcal{SHIF}$ Tableau

In this section, we show how the tableaux algorithm for $\mathcal{SHI}$ can be extended to deal with $\mathcal{SHIF}$-concepts. The following is a list of modifications that are necessary to deal with functional roles. The resulting definitions are then given in Definition 10.

1. If a node x has more R-neighbours than allowed by a functional restriction $(\leqslant 1\ R)$, we will merge these R-neighbours into a single one. Since these

R-neighbours can also be neighbours with respect to some roles S, S' which are not comparable by $\sqsubseteq^*$, the merged R-neighbour is also an S- and an S'-neighbour of x. To capture this, edges will be labelled with *sets* of roles.

2. Due to the new, set-valued edge labelling, the definitions of neighbours and successors have to be adjusted; as described in Definition 10.

3. The blocking strategy from Section 2.3 is extended by using pair-wise blocking as described in Section 4.1.

4. Tableau expansion rules must be added for functional restriction concepts, and the $\exists$-rule must be amended in order to deal with set valued edge labels. The complete set of $\mathcal{SHIF}$ expansion rules is given in Figure 4. For the proof of the soundness of these rules, namely the proof of Lemma 12, if $(\geqslant 2\ R) \in \mathcal{L}(x)$, then we always introduce two R-successors which can never be merged. For this purpose, we use a concept name A that does not occur in the input concept D and thus does not interfere with the other constraints. For implementation purposes, this rule could clearly be simplified,[2] but its current design facilitates the proofs.

5. The definition of a *clash* is extended to include those cases where there are conflicting functional restrictions. Given the $\geqslant$-rule as described, this is not strictly necessary, but it would be required if the $\geqslant$-rule did not create two logically disjoint successors.

Definition 10 In contrast to completion trees introduced in the previous Sections, in the following, each edge of completion trees is labelled with a *set* of roles.

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

A node x is *directly blocked* if none of its ancestors are blocked, and it has ancestors x', y and y' such that

1. x is a successor of x' and y is a successor of y' *and*
2. $\mathcal{L}(x) = \mathcal{L}(y)$ and $\mathcal{L}(x') = \mathcal{L}(y')$ *and*
3. $\mathcal{L}(\langle x', x\rangle) = \mathcal{L}(\langle y', y\rangle)$.

In this case we will say that y blocks x.

A node is *indirectly blocked* if its predecessor is blocked, and in order to avoid wasted expansion after an application of the $\leqslant$-rule, a node y will also be taken to be indirectly blocked if it is a successor of a node x and $\mathcal{L}(\langle x, y\rangle) = \emptyset$.

[2] It is intuitively obvious that if $(\geqslant 2\ R) \in \mathcal{L}(x)$, and there is no conflicting functional restriction in $\mathcal{L}(x)$, then the sub-tree rooted in a single R-successor of x could be duplicated in order to satisfy $(\geqslant 2\ R)$.

$\sqcap$-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.	$\forall S.C \in \mathcal{L}(x)$, x is not indirectly blocked,
		2.	there is some R with $\mathsf{Trans}(R)$ and $R \sqsubseteq^* S$, and
		3.	there is an R-neighbour y of x with $\forall R.C \notin \mathcal{L}(y)$
	then		$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\forall R.C\}$
$\geqslant$-rule:	if	1.	$(\geqslant 2\ R) \in \mathcal{L}(x)$, x is not blocked, and
		2.	there is no R-neighbour y of x with $A \in \mathcal{L}(y)$
	then		create two new nodes y_1, y_2 with $\mathcal{L}(\langle x, y_1\rangle) = \{R\}$, $\mathcal{L}(\langle x, y_2\rangle) = \{R\}$, $\mathcal{L}(y_1) = \{A\}$ and $\mathcal{L}(y_2) = \{\neg A\}$
$\leqslant$-rule:	if	1.	$(\leqslant 1\ R) \in \mathcal{L}(x)$, x is not indirectly blocked,
		2.	x has two R-neighbours y and z s.t. y is not an ancestor of z,
	then		1. $\mathcal{L}(z) \longrightarrow \mathcal{L}(z) \cup \mathcal{L}(y)$ and
			2. if z is an ancestor of y
			then $\mathcal{L}(\langle z, x\rangle) \longrightarrow \mathcal{L}(\langle z, x\rangle) \cup \mathsf{Inv}(\mathcal{L}(\langle x, y\rangle))$
			else $\mathcal{L}(\langle x, z\rangle) \longrightarrow \mathcal{L}(\langle x, z\rangle) \cup \mathcal{L}(\langle x, y\rangle)$
			3. $\mathcal{L}(\langle x, y\rangle) \longrightarrow \emptyset$

Figure 4: The complete tableaux expansion rules for $\mathcal{SHIF}$

For a node x, $\mathcal{L}(x)$ is said to contain a *clash* if it contains an $\mathcal{SHI}$-clash, or, for roles R and S, $\{(\leqslant 1\ R), (\geqslant 2\ S)\} \subseteq \mathcal{L}(x)$ and $S \sqsubseteq^* R$.

4.3 Soundness and Completeness

The soundness and completeness proof follows the same pattern as those for the other logics, but the tableaux construction proof is more complex as it must be able to create an infinite tableau.

Lemma 11 *For each $\mathcal{SHIF}$-concept D, the tableaux algorithm terminates.*

Proof: Very similar to the proof of Lemma 2, it only being necessary to show that there are a finite number of different node-relation-node triples. Let $m = |sub(D)|$ and $n = |\mathbf{R}_D|$. 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. Edge labels can only be changed by the $\leqslant$-rule which either expands them or sets them to $\emptyset$; in the latter case the node below the $\emptyset$-labelled edge is blocked and will remain blocked forever.

2. Nodes are labelled with nonempty subsets of $sub(D) \cup \{A, \neg A\}$ and edges with subsets of R_D, so there are at most 2^{2mn} different possible labellings for a pair of nodes and an edge. Therefore, if a path p is of length at least 2^{2mn}, then, from the pair-wise blocking condition defined in Section 4.2, there must be 2 nodes x, y on p such that x is directly blocked by y. Since a path on which nodes are blocked cannot become longer, paths are of length at most 2^{2mn}. This implies that also the maximum distance of any node to root node is bounded by 2^{2mn}.

3. Successors of a node x must be the result of an application of the $\exists$- or the $\geqslant$-rule to concepts of the form $\exists R.C$ and $(\geqslant 2\ R)$ in $\mathcal{L}(x)$. Each of these concepts triggers the generation of at most two successors y: note that if the $\leqslant$-rule subsequently causes $\mathcal{L}(\langle x, y\rangle)$ to be changed to $\emptyset$, then x will have some R-neighbour z with $\mathcal{L}(z) \supseteq \mathcal{L}(y)$. This, together with the enhanced definition of a clash, implies that the rule application which led to the generation of y will not be repeated. Since $sub(D)$ contains a total of at most m $\exists R.C$ and $(\geqslant 2\ R)$ concepts, the out-degree of the tree is bounded by $2m$. ∎

Lemma 12 (Soundness) *If the expansion rules can be applied to an $\mathcal{SHIF}$-concept D such that they yield a complete and clash-free completion tree, then D has a tableau.*

Proof: Intuitively, the definition of a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ from a complete and clash-free completion tree $\mathbf{T}$ works as follows: An individual in $\mathbf{S}$ corresponds to a *path* in $\mathbf{T}$ from the root node to some node that is not blocked. To obtain infinite tableaux, these paths may be *cyclic*. Instead of going to a directly blocked node, these paths go "back" to the blocking node—and this an infinite number of times. Thus, if blocking occurred while constructing a tableaux, we obtain an infinite tableau. [3]

More precisely, let $\mathbf{T}$ be a complete and clash-free completion tree. We will use the mapping $\mathsf{Tail}(p)$ to return the last element in a path p: given a path $p = [x_0, \ldots, x_n]$, where the x_i are nodes in $\mathbf{T}$, $\mathsf{Tail}(p) = x_n$. Paths in $\mathbf{T}$ are defined inductively as follows:

[3] If a simplified $\geqslant$-rule were employed, as outlined in Section 4.2, then a more elaborate construction would be required, one that created duplicate paths as necessary in order to satisfy $(\geqslant 2\ R)$ concepts.

1. For the root node x_0 in $\mathbf{T}$, $[x_0]$ is a path in $\mathbf{T}$;
2. For a path p and a node x_i in $\mathbf{T}$, $[p, x_i]$ is a path in $\mathbf{T}$ iff
   (a) x_i is a successor of $\mathsf{Tail}(p)$ and x_i is not blocked, *or*
   (b) for some node y in $\mathbf{T}$, y is a successor of $\mathsf{Tail}(p)$ and x_i blocks y.

Now we can define a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ with:

$$\begin{aligned}
\mathbf{S} &= \{x_p \mid p \text{ is a path in } \mathbf{T}\} \\
\mathcal{L}(x_p) &= \mathcal{L}(\mathsf{Tail}(p)) \\
\mathcal{E}(R) &= \{\langle x_p, x_q\rangle \in \mathbf{S} \times \mathbf{S} \mid \text{Either } q = [p, \mathsf{Tail}(q)] \text{ and} \\
&\qquad 1.\ \mathsf{Tail}(q) \text{ is an } R\text{-successor of } \mathsf{Tail}(p), \textit{ or} \\
&\qquad 2.\ \text{for some node } y \text{ in } \mathbf{T},\ y \text{ is an } R\text{-successor} \\
&\qquad\quad \text{of } \mathsf{Tail}(p) \text{ and } \mathsf{Tail}(q) \text{ blocks } y \\
&\qquad \text{or } p = [q, \mathsf{Tail}(p)] \text{ and} \\
&\qquad 1.\ \mathsf{Tail}(p) \text{ is an } \mathsf{Inv}(R)\text{-successor of } \mathsf{Tail}(q), \textit{ or} \\
&\qquad 2.\ \text{for some node } y \text{ in } \mathbf{T},\ y \text{ is an } \mathsf{Inv}(R)\text{-successor} \\
&\qquad\quad \text{of } \mathsf{Tail}(q) \text{ and } \mathsf{Tail}(p) \text{ blocks } y\}
\end{aligned}$$

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 $[x_0]$ is a path in $\mathbf{T}$. Hence $D \in \mathcal{L}(x_{[x_0]})$ for $x_{[x_0]} \in \mathbf{S}$.
2. The proof that Properties 1–3 of Definition 2 are satisfied is identical to the proof of Lemma 3.
3. The proof that Properties 4–6 of Definition 2 are satisfied is similar to that given in the proof of Lemma 3, with the additional observations that
   (a) The new Definition 10 of R-neighbours must be taken into account.
   (b) For all individuals $x_p \in \mathbf{S}$, the "immediate environment" of x_p is identical to that of the node $\mathsf{Tail}(p)$ in $\mathbf{T}$. To be more precise, for all nodes x in $\mathbf{T}$, if y is an R-neighbour of x, then, for every individual $x_p \in \mathbf{S}$ with $\mathsf{Tail}(p) = x$, there is an individual $x_q \in \mathbf{S}$ such that $\langle x_p, x_q\rangle \in \mathcal{E}(R)$ and $\mathcal{L}(x_q) = \mathcal{L}(y)$. This is straightforward in the case where y is an R-successor of x: either $\mathsf{Tail}(q) = y$, or $\mathsf{Tail}(q) = z$ for some z that blocks y and $\mathcal{L}(z) = \mathcal{L}(y)$.

   However, in the case where x is an $\mathsf{Inv}(R)$-successor of y (so y is an R-neighbour of x), and x blocks some node z, the maintenance of this property crucially depends on the definition of pair-wise blocking: let w be the predecessor of z, q be a path with $\mathsf{Tail}(q) = w$ and p be the path $[q, x]$ resulting from the block. By definition $x_p \in \mathbf{S}$ with $\mathsf{Tail}(p) = x$. From pair-wise blocking we have that w is an R-neighbour of z, and $\mathcal{L}(w) = \mathcal{L}(y)$, so for $x_q \in \mathbf{S}$, $\langle x_p, x_q\rangle \in \mathcal{E}(R)$ and $\mathcal{L}(x_q) = \mathcal{L}(y)$.

4. Property 7 holds because of the symmetric definition of the mapping $\mathcal{E}$.

5. The proof that Properties $6'$ and 8 of Definition 5 are satisfied is identical to the proof of Lemma 7.

6. Suppose Property 9 of Definition 9 were not satisfied. Let $x_p, x_q, x_{q'}$ be individuals in $\mathbf{S}$ with $(\leqslant 1\ R) \in \mathcal{L}(x_p)$, $\{\langle x_p, x_q\rangle, \langle x_p, x_{q'}\rangle\} \subseteq \mathcal{E}(R)$ and $q \neq q'$. This means that either

   (a) $\mathsf{Tail}(q)$ and $\mathsf{Tail}(q')$ are both R-neighbours of $\mathsf{Tail}(p)$, *or*

   (b) one of them, say $\mathsf{Tail}(q)$, is an R-neighbour of $\mathsf{Tail}(p)$ and $\mathsf{Tail}(q')$ blocks an R-neighbour y of $\mathsf{Tail}(p)$, but then both y and $\mathsf{Tail}(q)$ are R-neighbours of $\mathsf{Tail}(p)$, and $y \neq \mathsf{Tail}(q)$ because y is blocked while $\mathsf{Tail}(q)$ is not, *or*

   (c) $\mathsf{Tail}(q)$ and $\mathsf{Tail}(q')$ block R-neighbours y and z of $\mathsf{Tail}(p)$, but then both y and z are R-neighbours of $\mathsf{Tail}(p)$, with $y \neq z$ because $q \neq q'$ and a blocked node has a unique blocking node.

   In all three cases $\mathsf{Tail}(p)$ has two R-neighbours, the $\leqslant$-rule would be applicable, and $\mathbf{T}$ cannot be complete.

7. Property 10 of Definition 9 follows immediately from the $\geqslant$-rule and the definition of the tableau T: since $\mathbf{T}$ is clash-free, if $(\geqslant 2\ R) \in \mathcal{L}(x_p)$, then $\mathsf{Tail}(p)$ in $\mathbf{T}$ has two R-successors y, z that cannot be blocked by the same node since, w.l.o.g. $A \in \mathcal{L}(y)$ and $\neg A \in \mathcal{L}(z)$. Hence $\langle x_p, x_q\rangle \in \mathcal{E}(R)$ and $\langle x_p, x_{q'}\rangle \in \mathcal{E}(R)$, with $q \neq q'$. ∎

Lemma 13 (Completeness) *If D has a tableau, then the expansion rules can be applied to an $\mathcal{SHIF}$-concept D such that they yield a complete and clash-free completion tree.*

Again, this proof is similar to the one for Lemma 4 and $\mathcal{SI}$-concepts. This is due to the fact that we did not introduce new non-deterministic rules. Given a tableau, we can trigger the application of the expansion rules such that they yield a complete and clash-free completion tree, with Properties 9 and 10 of Definition 9 ensuring that it is also complete and clash-free with respect to functional restrictions.

Theorem 4 *The tableaux algorithm is a decision procedure for the satisfiability and subsumption of $\mathcal{SHIF}$-concepts with respect to terminologies.*

Using the same techniques and arguments as in Section 3.1, general concept inclusion axioms can be internalised. Hence the tableaux algorithm is a decision procedure for satisfiability and subsumption of $\mathcal{ALCFI}_{R^+}$-concepts with respect to terminologies.

5 Extending $\mathcal{SHI}$ by Qualifying Number Restrictions

In the following we present, in detail, a tableaux algorithm for $\mathcal{SHIQ}$, which extends $\mathcal{SHIF}$ with qualifying number restrictions. The algorithm combines the methods already used in the previous section with techniques that are used to deal with qualifying number restrictions [HB91; BBH96; Tob99]. In order to avoid references to definitions and proofs in all previous sections, we make this section self-contained. Of course, this leads to various repetitions, yet, it increases the readability.

5.1 Syntax and Semantics

Definition 11 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 *$\mathcal{SHIQ}$-roles* is $\mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$. A *role inclusion axiom* is of the form

$$R \sqsubseteq S,$$

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

A role R is called *simple* with respect to $\mathcal{R}^+$ iff $R \notin \mathbf{R}_+$ and, for any $S \circledast R$, S is also a simple role.

Let N_C be a set of *concept names*. The set of *$\mathcal{SHIQ}$-concepts* is the smallest set such that

1. every concept name $C \in N_C$ is a concept,
2. if C and D are concepts and R is an $\mathcal{SHIQ}$-role, then $(C \sqcap D)$, $(C \sqcup D)$, $(\neg C)$, $(\forall R.C)$, and $(\exists R.C)$ are concepts, and
3. if C is a concept, R is a simple $\mathcal{SHIQ}$-role and $n \in \mathbb{N}$, then $(\leqslant n\ R\ C)$ and $(\geqslant n\ R\ C)$ 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

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

where for a set M we denote the cardinality of M by $\sharp M$, $R^{\mathcal{I}}(x, C)$ is defined as $\{y \mid \langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$, and we define, for $P \in \mathbf{R}$ and $R \in \mathbf{R}_+$,

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

An interpretation $\mathcal{I}$ satisfies a role hierarchy $\mathcal{R}^+$, iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ for each $R \sqsubseteq S \in \mathcal{R}^+$; we denote this fact by $\mathcal{I} \models \mathcal{R}^+$.

A concept C is called *satisfiable with respect to a role hierarchy* $\mathcal{R}^+$ iff there is some interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{R}^+$ and $C^{\mathcal{I}} \neq \emptyset$. Such an interpretation is called a *model of* C with respect to $\mathcal{R}^+$. A concept D *subsumes* a concept C with respect to $\mathcal{R}^+$ (written $C \sqsubseteq_{\mathcal{R}^+} D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for each interpretation $\mathcal{I}$ with $\mathcal{I} \models \mathcal{R}^+$. Two concepts C, D are *equivalent* with respect to $\mathcal{R}^+$ (written $C \equiv_{\mathcal{R}^+} D$) 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 $x \in C^{\mathcal{I}}$.

5.2 An $\mathcal{SHIQ}$-Tableau

Again we define a satisfiability criterion that can be tested by a tableaux algorithm. 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 $\mathcal{SHIQ}$-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:

$$\begin{array}{rcl}
\neg(\exists R.C) & \equiv & (\forall R.\neg C) \\
\neg(\forall R.C) & \equiv & (\exists R.\neg C) \\
\neg(\leqslant n\ R\ C) & \equiv & (\geqslant (n+1)\ R\ C) \\
\neg(\geqslant n\ R\ C) & \equiv & \begin{cases} (\forall R.\neg C) & \text{if } n = 1 \\ (\leqslant (n-1)\ R\ C) & \text{otherwise} \end{cases}
\end{array}$$

For a concept C we will denote the NNF of $\neg C$ by $\sim C$.

Definition 12 For an $\mathcal{SHIQ}$-concept D in NNF we define $clos(D)$ to be the smallest set that contains D and is closed under sub-formulae and $\sim$.

Definition 13 If $\mathcal{R}^+$ is a role hierarchy, D is an $\mathcal{SHIQ}$-concept in NNF and $\mathbf{R}_D$ is the set of roles occurring in D, together with their inverses, a tableau T for D with respect to $\mathcal{R}^+$ is defined to be a triple $(\mathbf{S}, \mathcal{L}, \mathcal{E})$ such that: $\mathbf{S}$ is a set of individuals, $\mathcal{L} : \mathbf{S} \to 2^{clos(D)}$ maps each individual to a set of concepts which is a subset of $clos(D)$, $\mathcal{E} : \mathbf{R}_D \to 2^{\mathbf{S}\times\mathbf{S}}$ 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 $s \in \mathbf{S}$, $C, C_1, C_2 \in clos(D)$, and $R, S \in \mathbf{R}_D$, T must satisfy:

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 $\forall S.C \in \mathcal{L}(s)$ and $\langle s,t\rangle \in \mathcal{E}(S)$, then $C \in \mathcal{L}(t)$,
5. if $\exists S.C \in \mathcal{L}(s)$, then there is some $t \in \mathbf{S}$ such that $\langle s,t\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)$,
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 $\forall R.C \in \mathcal{L}(t)$,
7. $\langle x,y\rangle \in \mathcal{E}(R)$ iff $\langle y,x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$,
8. if $\langle s,t\rangle \in \mathcal{E}(R)$ and $R \sqsubseteq^* S$ then $\langle s,t\rangle \in \mathcal{E}(S)$,
9. if $(\leqslant n\ S\ C) \in \mathcal{L}(s)$, then $\sharp S^T(s,C) \leqslant n$,
10. if $(\geqslant n\ S\ C) \in \mathcal{L}(s)$, then $\sharp S^T(s,C) \geqslant n$,
11. if $(\bowtie\ n\ S\ C) \in \mathcal{L}(s)$ and $\langle s,t\rangle \in \mathcal{E}(S)$ then $C \in \mathcal{L}(t)$ or $\sim C \in \mathcal{L}(t)$,

where we use $\bowtie$ as a placeholder for both $\leqslant$ and $\geqslant$ and we define

$$S^T(s,C) := \{t \in \mathbf{S} \mid \langle s,t\rangle \in \mathcal{E}(S) \text{ and } C \in \mathcal{L}(t)\}.$$

Lemma 14 *An $\mathcal{SHIQ}$-concept D is satisfiable with respect to a role hierarchy $\mathcal{R}^+$ iff there exists a tableau for D with respect to $\mathcal{R}^+$.*

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 5. 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:

$$\begin{array}{lcl} \Delta^{\mathcal{I}} & = & \mathbf{S} \\ A^{\mathcal{I}} & = & \{s \mid A \in \mathcal{L}(s)\} \text{ for all concept names A in } \mathit{clos}(D) \\ R^{\mathcal{I}} & = & \begin{cases} \mathcal{E}(R)^+ & \text{if } \mathsf{Trans}(R) \\ \mathcal{E}(R) \cup \bigcup_{P \sqsubseteq^* R, P \neq R} P^{\mathcal{I}} & \text{otherwise} \end{cases} \end{array}$$

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 $\langle s,t\rangle \in S^{\mathcal{I}}$, then either $\langle s,t\rangle \in \mathcal{E}(S)$ 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 $R \sqsubseteq^* S$.

To show that $\mathcal{I}$ is a model of D w.r.t. $\mathcal{R}^+$ we have to prove (1) $\mathcal{I} \models \mathcal{R}^+$ and (2) $D^{\mathcal{I}} \neq \emptyset$. The first part is obvious due to Property 7 of Def. 13. The second part is shown by proving $\mathcal{I} \models \mathcal{R}^+$ and $C \in \mathcal{L}(s) \Rightarrow s \in C^{\mathcal{I}}$ for any $s \in \mathbf{S}$. This implies $D^{\mathcal{I}} \neq \emptyset$ since T is a tableau for D and hence there must be some $s \in \mathbf{S}$ with $D \in \mathcal{L}(s)$. It is not longer possible to use induction over the length of the

concept C, instead we will use the following *norm* $\|\cdot\|$ of a concept C. The norm $\|C\|$ for concept in NNF is inductively defined by:

$$\begin{array}{llllll} \|A\| & := & \|\neg A\| & := & 0 & \text{for } A \in N_C \\ \|C_1 \sqcap C_2\| & := & \|C_1 \sqcup C_2\| & := & 1 + \|C_1\| + \|C_2\| & \\ \|\forall R.C\| & := & \|\exists R.C\| & := & 1 + \|C\| & \\ \|(\bowtie\ n\ S\ C)\| & & & := & 1 + \|C\| & \end{array}$$

The two base cases of the induction are $C = A$ or $C = \neg A$. If $A \in \mathcal{L}(s)$, then by definition $s \in A^{\mathcal{I}}$. If $\neg A \in \mathcal{L}(s)$, then by Property 1, $A \notin \mathcal{L}(s)$ and hence $s \notin A^{\mathcal{I}}$. For the induction step we have to distinguish several cases:

- $C = C_1 \sqcap C_2$. Since T is a tableau, $C \in \mathcal{L}(s)$ implies $C_1 \in \mathcal{L}(s)$ and $C_2 \in \mathcal{L}(s)$. Hence, by induction, we have $s \in C_1^{\mathcal{I}}$ and $s \in C_2^{\mathcal{I}}$ which yields $s \in (C_1 \sqcap C_2)^{\mathcal{I}}$.

- $C = C_1 \sqcup C_2$. Similar to the previous case.

- $C = \exists S.E$. Since T is a tableau, $C \in \mathcal{L}(s)$ implies the existence of an individual $t \in \mathbf{S}$ such that $\langle s, t\rangle \in \mathcal{E}(S)$ and $E \in \mathcal{L}(t)$. By induction, we have $t \in E^{\mathcal{I}}$ and from the definition of $S^{\mathcal{I}}$ and Property 7 it follows that $\langle s, t\rangle \in S^{\mathcal{I}}$ and hence $s \in C^{\mathcal{I}}$.

- $C = \forall S.E$. Let $s \in \mathbf{S}$ with $C \in \mathcal{L}(s)$, let $t \in \mathbf{S}$ be an arbitrary individual such that $\langle s, t\rangle \in S^{\mathcal{I}}$. There are two possibilities:

  - $\langle s, t\rangle \in \mathcal{E}(S)$. Then Property 4 implies $E \in \mathcal{L}(t)$ and by induction $t \in E^{\mathcal{I}}$.
  - $\langle s, t\rangle \notin \mathcal{E}(S)$. Then 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 $R \sqsubseteq^* S$. Then Property 6 implies $\forall R.E \in \mathcal{L}(s_i)$ for all $1 \leq i \leq n$ and, from Property 4, $E \in \mathcal{L}(t)$ also holds. Again, by induction, this implies $t \in E^{\mathcal{I}}$.

  In both cases we have $t \in E^{\mathcal{I}}$ and since t has been chosen arbitrarily, $s \in C^{\mathcal{I}}$ holds.

- $C = (\geqslant n\ S\ E)$. For an s with $C \in \mathcal{L}(s)$ we have $\sharp S^T(s, E) \geqslant n$. Hence there are n individuals $t_1, \ldots, t_n$ such that $t_i \neq t_j$ for $i \neq j$, $\langle s, t_i\rangle \in \mathcal{E}(S)$, and $E \in \mathcal{L}(t_i)$ for all i. By induction, we have $t_i \in E^{\mathcal{I}}$ and, since $\mathcal{E}(S) \subseteq S^{\mathcal{I}}$, also $s \in C^{\mathcal{I}}$.

- $C = (\leqslant m\ S\ E)$. For this case we need that S is a simple role, which implies $S^{\mathcal{I}} = \mathcal{E}(S)$. Let s be an individual with $C \in \mathcal{L}(s)$. Due to Property 11 we have $E \in \mathcal{L}(t)$ or $\sim E \in \mathcal{L}(t)$ for each t with $\langle s, t\rangle \in \mathcal{E}(S)$. Moreover, $\sharp S^T(s, E) \leqslant n$ holds due to Property 9. We can show that $\sharp S^{\mathcal{I}}(s, E) \leqslant \sharp S^T(s, E)$: assume $\sharp S^{\mathcal{I}}(s, E) > \sharp S^T(s, E)$. This implies the existence of some t with $\langle s, t\rangle \in S^{\mathcal{I}}$ with $t \in E^{\mathcal{I}}$ but $E \notin \mathcal{L}(t)$ (because $S^{\mathcal{I}} = \mathcal{E}(S)$). By Property 11 this implies $\sim E \in \mathcal{L}(t)$, which, by induction yields $t \in (\sim E)^{\mathcal{I}}$ in contradiction to $t \in E^{\mathcal{I}}$.

For the *only-if* direction we have to show that satisfiability of D with respect to $\mathcal{R}^+$ implies the existence of a tableau T for D with respect to $\mathcal{R}^+$.

Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a model of D with $\mathcal{I} \models \mathcal{R}^+$. A tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ for D can be defined as:

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

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

- The Properties 1–5, 7, and 9–11 are satisfied as a direct consequence of the definition of the semantics of $\mathcal{SHIQ}$-concepts.
- If $s \in (\forall S.C)^{\mathcal{I}}$ and $\langle s,t\rangle \in R^{\mathcal{I}}$ for R with $\mathsf{Trans}(R)$ and $R \sqsubseteq^* S$, then $t \in (\forall R.C)^{\mathcal{I}}$ unless there is some u such that $\langle t,u\rangle \in R^{\mathcal{I}}$ and $u \notin C^{\mathcal{I}}$. In this case, if $\langle s,t\rangle \in R^{\mathcal{I}}$, $\langle t,u\rangle \in R^{\mathcal{I}}$ and $\mathsf{Trans}(R)$, then $\langle s,u\rangle \in R^{\mathcal{I}}$. Hence $\langle s,u\rangle \in S^{\mathcal{I}}$ and $s \notin (\forall S.C)^{\mathcal{I}}$—in contradiction to the assumption. T therefore satisfies Property 6.
- Property 8 is satisfied because $\mathcal{I} \models \mathcal{R}^+$. ∎

5.3 Constructing an $\mathcal{SHIQ}$-Tableau

From Lemma 14, an algorithm which constructs a tableau for an $\mathcal{SHIQ}$-concept D can be used as a decision procedure for the satisfiability of D with respect to a role hierarchy $\mathcal{R}^+$. Such an algorithm will now be described in detail. It uses the same techniques used for $\mathcal{SHIF}$, especially the pairwise-blocking. Additionally, we add rules to deal with the qualifying number restrictions. It is not sufficient to add a $\geqslant$- and $\leqslant$-rule similar to the ones from the $\mathcal{SHIF}$-algorithm. In order for the algorithm to be correct, we also have to make sure that certain *relevant* constraints are always added to labels of certain nodes; this is ensured by the *choose*-rule. In order to guarantee the termination of the algorithm, we have to make sure that the $\geqslant$- and $\leqslant$-rules can not be applied in a way that would yield to an infinite sequence of rule applications. This is enforced by recording which nodes have been introduced by an application of the $\geqslant$-rule and by prohibiting an identification of these nodes by the $\leqslant$-rule.

Definition 14 Let $\mathcal{R}^+$ be a role hierarchy. A *completion tree* with respect to $\mathcal{R}^+$ is a tree $\mathbf{T}$ where each node x of the tree is labelled with a set $\mathcal{L}(x) \subseteq \mathit{clos}(D)$ and each edge $\langle x,y\rangle$ is labelled with a set of role names $\mathcal{L}(\langle x,y\rangle)$ containing (possibly inverse) roles occurring in $\mathit{clos}(D)$. Additionally, we keep track of inequalities between nodes of the tree with a symmetric binary relation $\neq$ between the nodes of $\mathbf{T}$.

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

For a role S, a concept C and a node x in $\mathbf{T}$ we define $S^{\mathbf{T}}(x, C)$ by

$$S^{\mathbf{T}}(x, C) := \{y \mid y \text{ is } S\text{-neighbour of } x \text{ and } C \in \mathcal{L}(y)\}$$

A node x is *directly blocked* if none of its ancestors are blocked, and it has ancestors x', y and y' such that

1. x is a successor of x' and y is a successor of y' *and*
2. $\mathcal{L}(x) = \mathcal{L}(y)$ and $\mathcal{L}(x') = \mathcal{L}(y')$ *and*
3. $\mathcal{L}(\langle x', x\rangle) = \mathcal{L}(\langle y', y\rangle)$.

In this case we will say that y blocks x.

A node is *indirectly blocked* if its predecessor is blocked, and in order to avoid wasted expansion after an application of the $\leqslant$-rule, a node y will also be taken to be indirectly blocked if it is a successor of a node x and $\mathcal{L}(\langle x, y\rangle) = \emptyset$.

For a node x, $\mathcal{L}(x)$ is said to contain a *clash* if, for some concept name $A \in N_C$, $\{A, \neg A\} \subseteq \mathcal{L}(x)$, or if for a some concept C, some role S, and some $n \in \mathbb{N}$: $(\leqslant\ n\ S\ C) \in \mathcal{L}(x)$ and there are $n+1$ nodes $y_0, \ldots, y_n$ such that $C \in \mathcal{L}(y_i)$, y_i is an S-neighbour of x and $y_i \neq y_j$ for all $0 \leq i < j \leq n$.

The algorithm initialises the 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. The inequality relation $\neq$ is initialised with the empty set. $\mathbf{T}$ is then expanded by repeatedly applying the rules from Figure 5.3.

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.

5.4 Soundness and Completeness

Again we will show that the algorithm is terminating, sound, and complete.

Lemma 15 *For each $\mathcal{SHIQ}$-concept D and role hierarchy $\mathcal{R}^+$, the tableaux algorithm terminates.*

Proof: This proof uses the same ideas as the proof of Lemma 11. Let $m = |clos(D)|$, $k = |\mathbf{R}_D|$, and n_{max} the maximum n that occurs in a concept of the form $(\bowtie\ n\ S\ C) \in clos(D)$. 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. Edge labels can only be changed by the $\leqslant$-rule which either expands them or sets them to $\emptyset$; in the latter case the node below the $\emptyset$-labelled edge is blocked and will remain blocked forever.

$\sqcap$-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.	$\forall S.C \in \mathcal{L}(x)$, 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	$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\forall R.C\}$
choose-rule:	if 1.	$(\bowtie\ n\ S\ C) \in \mathcal{L}(x)$, x is not indirectly blocked, and
	2.	there is an S-neighbour y of x with $\{C, \sim C\} \cap \mathcal{L}(y) = \emptyset$
	then	$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{E\}$ for some $E \in \{C, \sim C\}$
$\geqslant$-rule:	if 1.	$(\geqslant n\ S\ C) \in \mathcal{L}(x)$, x is not blocked, and
	2.	there are no n nodes $y_1, \ldots, y_n$ such that $C \in \mathcal{L}(y_i)$, y_i is an S-neighbour of x, and $y_i \neq y_j$ for $1 \leq i < j \leq n$
	then	create n new nodes $y_1, \ldots, y_n$ with $\mathcal{L}(\langle x, y_i\rangle) = \{S\}$, $\mathcal{L}(y_i) = \{C\}$, and $y_i \neq y_j$ for $1 \leq i < j \leq n$.
$\leqslant$-rule:	if 1.	$(\leqslant n\ S\ C) \in \mathcal{L}(x)$, x is not indirectly blocked, and
	2.	$\sharp S^{\mathbf{T}}(x, C) > n$ and there are two S-neighbours y, z of x with $C \in \mathcal{L}(y), C \in \mathcal{L}(z)$, y is not an ancestor of z, and not $y \neq z$
	then	1. $\mathcal{L}(z) \longrightarrow \mathcal{L}(z) \cup \mathcal{L}(y)$ and 2. if z is an ancestor of y then $\mathcal{L}(\langle z, x\rangle) \longrightarrow \mathcal{L}(\langle z, x\rangle) \cup \mathsf{Inv}(\mathcal{L}(\langle x, y\rangle))$ else $\mathcal{L}(\langle x, z\rangle) \longrightarrow \mathcal{L}(\langle x, z\rangle) \cup \mathcal{L}(\langle x, y\rangle)$ 3. $\mathcal{L}(\langle x, y\rangle) \longrightarrow \emptyset$ 4. Set $u \neq z$ for all u with $u \neq y$

Figure 5: The Completion Rules for $\mathcal{SHIQ}$

2. Successors of a node x must be the result of an application of the $\exists$- or the $\geqslant$-rule to concepts of the form $\exists R.C$ (which yields one successor) and $(\geqslant n\ S\ C)$ (which yields n successors) in $\mathcal{L}(x)$. For a node x, each of these concepts can trigger the generation of successors at most once. For the $\exists$-rule, if a successor y of x was generated for a concept $\exists S.C \in \mathcal{L}(x)$ and later $\mathcal{L}(\langle x, y\rangle)$ is set to $\emptyset$ by an application of the $\leqslant$-rule, then there will be some S-neighbour z of x such that $C \in \mathcal{L}(z)$. For the $\geqslant$-rule: If $y_1, \ldots, y_n$ were generated by an application of the $\geqslant$-rule for a concept $(\geqslant n\ S\ C)$, then $y_i \not\approx y_j$ holds for all $1 \leq i < j \leq n$. This implies that there will always be n S-neighbours $y'_1, \ldots, y'_n$ of x with $C \in \mathcal{L}(y'_i)$ and $y'_i \not\approx y'_j$ for all $1 \leq i < j \leq n$, since the $\leqslant$-rule can never merge two nodes y'_i, y'_j (because $y'_i \not\approx y'_j$), and, whenever an application of the $\leqslant$-rule sets $\mathcal{L}(\langle x, y'_i\rangle)$ to $\emptyset$, then there will be some S-neighbour z of x with $C \in \mathcal{L}(z)$ and z "inherits" all inequalities from y'_i.

   Since $clos(D)$ contains a total of at most m $\exists R.C$ and $(\geqslant n\ S\ C)$ concepts, the out-degree of the tree is bounded by $m \cdot n_{max}$.

3. Nodes are labelled with nonempty subsets of $clos(D)$ and edges with subsets of R_D, so there are at most 2^{2mk} different possible labellings for a pair of nodes and an edge. Therefore, if a path p is of length at least 2^{2mk}, then from the pair-wise blocking condition defined in Definition 14 there must be two nodes x, y on p such that x is directly blocked by y. Since a path on which nodes are blocked cannot become longer, paths are of length at most 2^{2mn}. ∎

Lemma 16 (Soundness) *If the expansion rules can be applied to an $\mathcal{SHIQ}$-concept D such that they yield a complete and clash-free completion tree with respect to $\mathcal{R}^+$, then D has a tableau with respect to $\mathcal{R}^+$.*

Proof: We use a path construction similar to the one in the proof of Lemma 12, however, due to qualifying number restrictions we have to take special care. In particular, we must distinguish different nodes that are blocked by the same node.

Let $\mathbf{T}$ be a complete and clash-free completion tree. A path is a sequence of pairs of nodes of $\mathbf{T}$ of the form $p = [(x_0, x'_0), \ldots, (x_n, x'_n)]$. For such a path we define $\mathsf{Tail}(p) := x_n$ and $\mathsf{Tail}'(p) := x'_n$. With $[p|(x_{n+1}, x'_{n+1})]$ we denote the path $[(x_0, x'_0), \ldots, (x_n, x'_n), (x_{n+1}, x'_{n+1})]$. The set $\mathsf{Paths}(\mathbf{T})$ is defined inductively as follows:

- For the root node x_0 of $\mathbf{T}$, $[(x_0, x_0)] \in \mathsf{Paths}(\mathbf{T})$, and
- For a path $p \in \mathsf{Paths}(\mathbf{T})$ and a node z in $\mathbf{T}$:
  - if z is a successor of $\mathsf{Tail}(p)$ and z is not blocked, then $[p|(z, z)] \in \mathsf{Paths}(\mathbf{T})$, or
  - if, for some node y in $\mathbf{T}$, y is a successor of $\mathsf{Tail}(p)$ and z blocks y, then $[p|(z, y)] \in \mathsf{Paths}(\mathbf{T})$.

Please note that, due to the construction of Paths, for $p \in \mathsf{Paths}(\mathbf{T})$ with $p = [p'|(x, x')]$, x is not blocked, x' is blocked iff $x \neq x'$, and x' is never indirectly blocked. Furthermore, $\mathcal{L}(x) = \mathcal{L}(x')$ holds.

Now we can define a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ with:

$$\begin{aligned}
\mathbf{S} &= \mathsf{Paths}(\mathbf{T}) \\
\mathcal{L}(p) &= \mathcal{L}(\mathsf{Tail}(p)) \\
\mathcal{E}(R) &= \{\langle p, q\rangle \in \mathbf{S} \times \mathbf{S} \mid \text{ Either } q = [p|(x, x')] \text{ and} \\
&\qquad\qquad x' \text{ is an } R\text{-successor of } \mathsf{Tail}(p) \\
&\qquad\quad \text{or } p = [q|(x, x')] \text{ and} \\
&\qquad\qquad x' \text{ is an } \mathsf{Inv}(R)\text{-successor of } \mathsf{Tail}(q)\}.
\end{aligned}$$

Claim: T is a tableau for D with respect to $\mathcal{R}^+$.

We have to show that T satisfies all the properties from Definition 13.

- $D \in \mathcal{L}([(x_0, x_0)])$ since $D \in \mathcal{L}(x_0)$.
- **Property 1** holds because $\mathbf{T}$ is clash-free; **Properties 2,3** hold because $\mathsf{Tail}(p)$ is not blocked and $\mathbf{T}$ is complete.
- **Property 4**: Assume $\forall S.C \in \mathcal{L}(p)$ and $\langle p, q\rangle \in \mathcal{E}(S)$. If $q = [p|(x, x')]$, then x' is an S-successor of $\mathsf{Tail}(p)$ and thus $C \in \mathcal{L}(x')$ must hold (because the $\forall$-rule is not applicable). Since $\mathcal{L}(q) = \mathcal{L}(x) = \mathcal{L}(x')$ we have $C \in \mathcal{L}(q)$. If $p = [q|(x, x')]$, then x' is an $\mathsf{Inv}(S)$-successor of $\mathsf{Tail}(q)$ and thus $C \in \mathcal{L}(\mathsf{Tail}(q))$ must hold (because x' is not indirectly blocked and the $\forall$-rule is not applicable), hence $C \in \mathcal{L}(q)$.
- **Property 5**: Assume $\exists S.C \in \mathcal{L}(p)$. Define $x := \mathsf{Tail}(p)$. In $\mathbf{T}$ there must an S-neighbour y of x with $C \in \mathcal{L}(y)$, because the $\exists$-rule is not applicable. There are two possibilities:
  - y is a successor of x in $\mathbf{T}$. If y is not blocked, then $q := [p|(y, y)] \in \mathbf{S}$ and $\langle p, q\rangle \in \mathcal{E}(S)$ as well as $C \in \mathcal{L}(q)$. If y is blocked by some node z in $\mathbf{T}$, then $q := [p|(z, y)] \in \mathbf{S}$.
  - y is a predecessor of x. Again, there are two possibilities:
    - p is of the form $p = [q|(x, x')]$ with $\mathsf{Tail}(q) = y$.
    - p is of the form $p = [q|(x, x')]$ with $\mathsf{Tail}(q) = u \neq y$. x only has one predecessor in $\mathbf{T}$, hence u is not the predecessor of x. This implies $x \neq x'$, x blocks x' in $\mathbf{T}$, and u is the predecessor of x' due to the construction of Paths. Together with the definition of the blocking condition, this implies $\mathcal{L}(\langle u, x'\rangle) = \mathcal{L}(\langle y, x\rangle)$ as well as $\mathcal{L}(u) = \mathcal{L}(y)$ due to the pairwise blocking.

  In all three cases, $\langle p, q\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(q)$.
- **Property 6**: Assume $\forall S.C \in \mathcal{L}(p)$, $\langle p, q\rangle \in \mathcal{E}(R)$ for some $R \sqsubseteq^* S$ with $\mathsf{Trans}(R)$. If $q = [p|(x, x')]$, then x' is an R-successor of $\mathsf{Tail}(p)$ and thus

$\forall R.C \in \mathcal{L}(x')$ must hold (because otherwise the $\forall_+$-rule would be applicable). From $\mathcal{L}(q) = \mathcal{L}(x) = \mathcal{L}(x')$ it follows that $\forall R.C \in \mathcal{L}(q)$. If $p = [q|(x, x')]$, then x' is an $\mathsf{Inv}(R)$-successor of $\mathsf{Tail}(q)$ and hence $\mathsf{Tail}(q)$ is an R-neighbour of x'. Because x' is not indirectly blocked, this implies $\forall R.C \in \mathcal{L}(\mathsf{Tail}(q))$ and hence $\forall R.C \in \mathcal{L}(q)$.

- **Property 11**: Assume $(\bowtie\ n\ S\ C) \in \mathcal{L}(p)$, $\langle p, q\rangle \in \mathcal{E}(S)$. If $q = [p|(x, x')]$ then x' is an S-successor of $\mathsf{Tail}(p)$ and thus $\{C, \sim C\} \cap \mathcal{L}(x') \neq \emptyset$ must hold (since the *choose*-rule is not applicable). Since $\mathcal{L}(q) = \mathcal{L}(x) = \mathcal{L}(x')$ we have $\{C, \sim C\} \cap \mathcal{L}(q) \neq \emptyset$. If $p = [q|(x, x')]$, then x' is an $\mathsf{Inv}(S)$-successor of $\mathsf{Tail}(q)$ and thus $\{C, \sim C\} \cap \mathcal{L}(\mathsf{Tail}(q)) \neq \emptyset$ must hold (since x' is not indirectly blocked and the *choose*-rule is not applicable), hence $\{C, \sim C\} \cap \mathcal{L}(q) \neq \emptyset$.

- Assume **Property 9** does not hold. Hence there is some $p \in \mathbf{S}$ with $(\leqslant n\ S\ C) \in \mathcal{L}(p)$ and $\sharp S^T(p, C) > n$. We will show that this implies $\sharp S^{\mathbf{T}}(\mathsf{Tail}(p), C) > n$ which is a contradiction to either the clash-freeness or completeness of $\mathbf{T}$. Define $x := \mathsf{Tail}(p)$ and $P := S^T(p, C)$. Due to the assumption, we have $\sharp P > n$. We distinguish two cases:

  - P contains only paths of the form $q = [p|(y, y')]$. We claim that the function Tail' is injective on P. If we assume that there are two paths $q_1, q_2 \in P$ with $q_1 \neq q_2$ and $\mathsf{Tail}'(q_1) = \mathsf{Tail}'(q_2) = y'$ then this implies that q_1 is of the form $q_1 = [p|(y_1, y')]$ and q_2 is of the form $q_2 = [p|(y_2, y')]$ with $y_1 \neq y_2$. If y' is not blocked in $\mathbf{T}$, then $y_1 = y' = y_2$ holds, contradicting $y_1 \neq y_2$. If y' is blocked in $\mathbf{T}$, then both y_1 and y_2 block y', which implies $y_1 = y_2$, again a contradiction. Since Tail' is injective on P, it holds that $\sharp P = \sharp\, \mathsf{Tail}'(P)$. Also for each $y' \in \mathsf{Tail}'(P)$, y' is an S-successor of x and $C \in \mathcal{L}(y')$. This implies $\sharp S^{\mathbf{T}}(x, C) > n$.
  - P contains a path q where p is of the form $p = [q|(x, x')]$. Obviously, P may only contain one such path. As in the previous case, Tail' is an injective function on the set $P' := P \setminus \{q\}$, each $y' \in \mathsf{Tail}'(P')$ is an S-successor of x and $C \in \mathcal{L}(y')$ for each $y' \in \mathsf{Tail}'(P')$. To show that indeed $\sharp S^{\mathbf{T}}(x, C) > n$ holds, we have to prove the existence of a further S-neighbour u of x with $C \in \mathcal{L}(u)$ and $u \notin \mathsf{Tail}'(P')$. This will be "supplied" by $z := \mathsf{Tail}(q)$. We distinguish two cases:
    - $x = x'$. Hence x is not blocked. This implies that x is an $\mathsf{Inv}(S)$-successor of z in $\mathbf{T}$. Since $\mathsf{Tail}'(P')$ contains only successors of x we have that $z \notin \mathsf{Tail}'(P')$ and, by construction, z is an S-neighbour of x with $C \in \mathcal{L}(z)$.
    - $x \neq x'$. This implies that x' is blocked in $\mathbf{T}$ by x and that x' is an $\mathsf{Inv}(S)$-successor of z in $\mathbf{T}$. Due to the definition of pairwise-blocking this implies that x is an $\mathsf{Inv}(S)$-successor of some node u in $\mathbf{T}$ with $\mathcal{L}(u) = \mathcal{L}(z)$. Again, since $\mathsf{Tail}'(P')$ contains only

successors of x we have that $u \notin \mathsf{Tail}'(P')$ and, by construction, u is an S-neighbour of x and $C \in \mathcal{L}(u)$.

- **Property 10**: Assume $(\geqslant n\ S\ C) \in \mathcal{L}(p)$. This implies that there exist n individuals $y_1, \ldots, y_n$ in $\mathbf{T}$ such that each y_i is an S-neighbour of $\mathsf{Tail}(p)$ and $C \in \mathcal{L}(y_i)$. We claim that, for each of these individuals, there is a path q_i such that $\langle p, q_i \rangle \in \mathcal{E}(S)$, $C \in \mathcal{L}(q_i)$, and $q_i \neq q_j$ for all $1 \leq i < j \leq n$. Obviously, this implies $\sharp S^T(p, C) \geqslant n$. For each y_i there are three possibilities:
  - y_i is an S-successor of x and y_i is not blocked in $\mathbf{T}$. Then $q_i = [p|(y_i, y_i)]$ is a path with the desired properties.
  - y_i is an S-successor of x and y_i is blocked in $\mathbf{T}$ by some node z. Then $q_i = [p|(z, y_i)]$ is the path with the desired properties. Since the same z may block several of the y_js, it is indeed necessary to include y_i explicitly into the path to make them distinct.
  - x is an $\mathsf{Inv}(S)$-successor of y_i. There may be at most one such y_i. This implies that p is of the form $p = [q|(x, x')]$ with $\mathsf{Tail}(q) = y_i$. Again, q has the desired properties and, obviously, q is distinct from all other paths q_j.
- **Property 7** is satisfied due to the symmetric definition of $\mathcal{E}$. **Property 8** is satisfied due to the definition of R-successor that takes into account the role hierarchy $\sqsubseteq^*$. ∎

Lemma 17 (Completeness) *If an $\mathcal{SHIQ}$-concept D has a tableau with respect to $\mathcal{R}^+$, then the expansion rules can be applied to an D such that they yield a complete and clash-free completion tree with respect to $\mathcal{R}^+$.*

Proof: Let $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ be a tableau for D w.r.t. $\mathcal{R}^+$. We use this tableau to guide the application of the non-deterministic rules. To do this, we will inductively define a function π, mapping the individuals of the tree $\mathbf{T}$ to $\mathbf{S}$ such that, for each x, y in $\mathbf{T}$:

$$\left.\begin{array}{l} \mathcal{L}(x) \subseteq \mathcal{L}(\pi(x)) \\ \text{if } y \text{ is an } S\text{-neighbour of } x \text{ then } \langle \pi(x), \pi(y) \rangle \in \mathcal{E}(S) \\ x \not\doteq y \text{ implies } \pi(x) \neq \pi(y) \end{array}\right\} (*)$$

Claim: Let $\mathbf{T}$ be a completion-tree and π a function that satisfies $(*)$. If a rule is applicable to $\mathbf{T}$ then the rule is applicable to $\mathbf{T}$ in a way that yields a completion-tree $\mathbf{T}'$ and a function π' that satisfy $(*)$.

Let $\mathbf{T}$ be a completion-tree and π be a function that satisfies $(*)$. We have to consider the various rules.

- The $\sqcap$-rule: If $C_1 \sqcap C_2 \in \mathcal{L}(x)$, then $C_1 \sqcap C_2 \in \mathcal{L}(\pi(x))$. This implies $C_1, C_2 \in \mathcal{L}(\pi(x))$ due to Property 2 from Definition 13, and hence the rule can be applied without violating $(*)$.

- The $\sqcup$-rule: If $C_1 \sqcup C_2 \in \mathcal{L}(x)$, then $C_1 \sqcup C_2 \in \mathcal{L}(\pi(x))$. Since T is a tableau, Property 3 from Definition 13 implies $\{C_1, C_2\} \cap \mathcal{L}(\pi(x)) \neq \emptyset$. Hence the $\sqcup$-rule can add a concept $E \in \{C_1, C_2\}$ to $\mathcal{L}(x)$ such that $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ holds.

- The $\exists$-rule: If $\exists S.C \in \mathcal{L}(x)$, then $\exists S.C \in \mathcal{L}(\pi(x))$ and, since T is a tableau, Property 5 of Definition 13 implies that there is an element $t \in \mathbf{S}$ such that $\langle \pi(x), t \rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)$. The application of the $\exists$-rule generates a new variable y with $\mathcal{L}(\langle x, y \rangle = \{S\}$ and $\mathcal{L}(y) = \{C\}$. Hence we set $\pi' := \pi[y \mapsto t]$ which yields a function that satisfies $(*)$ for the modified tree.

- The $\forall$-rule: If $\forall S.C \in \mathcal{L}(x)$, then $\forall S.C \in \mathcal{L}(\pi(x))$, and if y is an S-neighbour of x, then also $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(S)$ due to $(*)$. Since T is a tableau, Property 4 of Definition 13 implies $C \in \mathcal{L}(\pi(y))$ and hence the $\forall$-rule can be applied without violating $(*)$.

- The $\forall_+$-rule: If $\forall S.C \in \mathcal{L}(x)$, then $\forall S.C \in \mathcal{L}(\pi(x))$, and if there is some $R \sqsubseteq S$ with $\mathsf{Trans}(R)$ and y is an R-neighbour of x, then also $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(R)$ due to $(*)$. Since T is a tableau, Property 6 of Definition 13 implies $\forall R.C \in \mathcal{L}(\pi(y))$ and hence the $\forall_+$-rule can be applied without violating $(*)$.

- The *choose*-rule: If $(\bowtie\ n\ S\ C) \in \mathcal{L}(x)$, then $(\bowtie\ n\ S\ C) \in \mathcal{L}(\pi(x))$, and, if there is an S-neighbour y of x, then $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(S)$ due to $(*)$. Since T is a tableau, Property 11 of Definition 13 implies $\{C, \sim C\} \cap \mathcal{L}(\pi(y) \neq \emptyset$. Hence the *choose*-rule can add an appropriate concept $E \in \{C, \sim C\}$ to $\mathcal{L}(x)$ such that $\mathcal{L}(y) \subseteq \mathcal{L}(\pi(y))$ holds.

- The $\geqslant$-rule: If $(\geqslant\ n\ S\ C) \in \mathcal{L}(x)$, then $(\geqslant\ n\ S\ C) \in \mathcal{L}(\pi(x))$. Since T is a tableau, Property 10 of Definition 13 implies $\sharp S^T(\pi(x), C) \geqslant n$. Hence there are individuals $t_1, \ldots, t_n \in \mathbf{S}$ such that $\langle \pi(x), t_i \rangle \in \mathcal{E}(S)$, $C \in \mathcal{L}(t_i)$, and $t_i \neq t_j$ for $1 \leq i < j \leq n$. The $\geqslant$-rule generates n new nodes $y_1, \ldots, y_n$. By setting $\pi' := \pi[y_1 \mapsto t_1, \cdots y_n \mapsto t_n]$, one obtains a function π' that satisfies $(*)$ for the modified tree.

- The $\leqslant$-rule: If $(\leqslant n\ S\ C) \in \mathcal{L}(x)$, then $(\leqslant n\ S\ C) \in \mathcal{L}(\pi(x))$. Since T is a tableau, Property 9 of Definition 13 implies $\sharp S^T(\pi(x), C) \leqslant n$. If the $\leqslant$-rule is applicable, we have $\sharp S^{\mathbf{T}}(x, C) > n$, which implies that there are at least $n+1$ S-neighbours $y_0, \ldots, y_n$ of x such that $C \in \mathcal{L}(y_i)$. Thus, there must be two nodes $y, z \in \{y_0, \ldots, y_n\}$ such that $\pi(y) = \pi(z)$ (because otherwise $\sharp S^T(\pi(x), C) > n$ would hold). $\pi(y) = \pi(z)$ implies that $y \not\doteq z$ cannot hold because of $(*)$, and y, z can be chosen such that y is not an ancestor of z. Hence the $\leqslant$-rule can be applied without violating $(*)$.

Why does this claim yield the completeness of the tableaux algorithm? For the initial completion-tree consisting of a single node x_0 with $\mathcal{L}(x_0) = \{D\}$ and $\not\doteq = \emptyset$ we can give a function π that satisfies $(*)$ by setting $\pi(x_0) := s_0$ for

some $s_0 \in \mathbf{S}$ with $D \in \mathcal{L}(s_0)$ (such an s_0 exists since T is a tableau for D). Whenever a rule is applicable to $\mathbf{T}$, it can be applied in a way that maintains $(*)$, and, from Lemma 15, we have that any sequence of rule applications must terminate. Since $(*)$ holds, any tree generated by these rule-applications must be clash-free. This can be seen as follows: There are two possibilities for a clash:

- $\mathbf{T}$ cannot contain a node x such that $\{C, \neg C\} \in \mathcal{L}(x)$ because $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ and hence Property 1 of Definition 13 would be violated for $\pi(x)$.
- $\mathbf{T}$ cannot contain a node x with $(\leqslant n\ S\ C) \in \mathcal{L}(x)$ and $n+1$ S-neighbours $y_0, \ldots y_n$ of x with $C \in \mathcal{L}(y_i)$ and $y_i \neq y_j$ for $0 \leq i < j \leq n$, because $(\leqslant n\ S\ C) \in \mathcal{L}(\pi(x))$, and, since $y_i \neq y_j$ implies $\pi(y_i) \neq \pi(y_j)$, $\sharp S^T(\pi(x), C) > n$ would hold which contradicts Property 9 of Definition 13. ∎

Since internalisation of terminologies is obviously also possible for $\mathcal{SHIQ}$ with the same techniques presented in Section 3.1, we have the following theorem:

Theorem 5 *The tableaux algorithm is a decision procedure for the satisfiability and subsumption of $\mathcal{SHIQ}$-concepts with respect to role hierarchies and terminologies.*

6 Deciding consistency of $\mathcal{SHIQ}$ ABoxes

In this section, we extend the completion algorithm presented in Section 5 to decide consistency of $\mathcal{SHIQ}$-ABoxes. This extension does not concern the expansion rules (besides a modification of the $\leqslant$-rule), but only the way in which the completion algorithm is started. We start with a definition of ABoxes.

Definition 15 Let $\mathbf{I} = \{a, b, c \ldots\}$ (possibly with subscripts) be a set of *individual* names. An *assertion* is of the form $a\!:\!C$, $(a, b)\!:\!R$, or $a \neq b$ for $a, b \in \mathbf{I}$, a (possibly inverse) role R, and a $\mathcal{SHIQ}$-concept C. An *ABox* is a finite set of assertions.

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is defined as in Definition 11, with the additional requirement that $\cdot^{\mathcal{I}}$ maps every individual a to an element $a^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$.[4]

An interpretation $\mathcal{I}$ satisfies an assertion $a\!:\!C$ (resp. $(a, b)\!:\!R$) iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$ (resp. $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$), and it satisfies an assertion $a \neq b$ iff $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. An interpretation satisfies an ABox iff it satisfies each assertion in the ABox. Such an interpretation is called a *model* of an ABox.

An ABox $\mathcal{A}$ is called *consistent* with respect to a role hierarchy $\mathcal{R}^+$ iff there is a model $\mathcal{I}$ of $\mathcal{A}$ such that $\mathcal{I} \models \mathcal{R}^+$. Such an interpretation is called a *model of* $\mathcal{A}$ w.r.t. $\mathcal{R}^+$.

[4] Please note that we do not employ the unique name assumption, i.e., we *do* allow two individual names to be interpreted as the same object. Instead, we have taken the more general approach of allowing explicit inequality assertions.

6.1 A Tableau for ABoxes

In the following, if not stated otherwise, C, D denote a $\mathcal{SHIQ}$-concepts in NNF, $\mathcal{R}^+$ a role hierarchy, $\mathbf{R}_\mathcal{A}$ the set of roles occurring in $\mathcal{A}$ and $\mathcal{R}^+$ together with their inverses, and $\mathbf{I}_\mathcal{A}$ is the set of individuals occurring in $\mathcal{A}$.

Without loss of generality, we assume all concepts C occurring in assertions $a_i{:}C \in \mathcal{A}$ to be in NNF.

Definition 16 $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{I})$ is a *tableau* for $\mathcal{A}$ w.r.t. $\mathcal{R}^+$ iff

- $\mathbf{S}$ is a set of objects,
- $\mathcal{L} : \mathbf{S} \to 2^{clos(\mathcal{A})}$ maps each object to a set of concepts,
- $\mathcal{E} : \mathbf{R}_D \to 2^{\mathbf{S}\times\mathbf{S}}$ maps each role to a set of pairs of individuals, and
- $\mathcal{I} : \mathbf{I}_\mathcal{A} \to \mathbf{S}$ maps individuals occurring in $\mathcal{A}$ to objects in $\mathbf{S}$.

Furthermore, for all $s, t \in \mathbf{S}$, $C, C_1, C_2 \in \mathit{clos}(\mathcal{A})$, and $R, S \in \mathbf{R}_\mathcal{A}$, it holds that:

(P1) if $C \in \mathcal{L}(s)$, then $\neg C \notin \mathcal{L}(s)$,

(P2) if $C_1 \sqcap C_2 \in \mathcal{L}(s)$, then $C_1 \in \mathcal{L}(s)$ and $C_2 \in \mathcal{L}(s)$,

(P3) if $C_1 \sqcup C_2 \in \mathcal{L}(s)$, then $C_1 \in \mathcal{L}(s)$ or $C_2 \in \mathcal{L}(s)$,

(P4) if $\forall S.C \in \mathcal{L}(s)$ and $\langle s, t\rangle \in \mathcal{E}(S)$, then $C \in \mathcal{L}(t)$,

(P5) if $\exists S.C \in \mathcal{L}(s)$, then there is some $t \in \mathbf{S}$ such that $\langle s, t\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)$,

(P6) 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 $\forall R.C \in \mathcal{L}(t)$,

(P7) $\langle x, y\rangle \in \mathcal{E}(R)$ iff $\langle y, x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$,

(P8) if $\langle s, t\rangle \in \mathcal{E}(R)$ and $R \sqsubseteq^* S$, then $\langle s, t\rangle \in \mathcal{E}(S)$,

(P9) if $(\leqslant n\ S\ C) \in \mathcal{L}(s)$, then $\sharp S^T(s, C) \leqslant n$,

(P10) if $(\geqslant n\ S\ C) \in \mathcal{L}(s)$, then $\sharp S^T(s, C) \geqslant n$,

(P11) if $(\bowtie\ n\ S\ C) \in \mathcal{L}(s)$ and $\langle s, t\rangle \in \mathcal{E}(S)$ then $C \in \mathcal{L}(t)$ or $\sim C \in \mathcal{L}(t)$,

(P12) if $a{:}C \in \mathcal{A}$, then $C \in \mathcal{L}(\mathcal{I}(a))$,

(P13) if $(a, b){:}R \in \mathcal{A}$, then $\langle \mathcal{I}(a), \mathcal{I}(b)\rangle \in \mathcal{E}(R)$,

(P14) if $a \neq b \in \mathcal{A}$, then $\mathcal{I}(a) \neq \mathcal{I}(b)$,

where $\bowtie$ is a placeholder for both $\leqslant$ and $\geqslant$, and $S^T(s, C) := \{t \in \mathbf{S} \mid \langle s, t\rangle \in \mathcal{E}(S)$ and $C \in \mathcal{L}(t)\}$.

Lemma 18 *A $\mathcal{SHIQ}$-ABox $\mathcal{A}$ is consistent w.r.t. $\mathcal{R}^+$ iff there exists a tableau for $\mathcal{A}$ w.r.t. $\mathcal{R}^+$.*

Proof: For the *if* direction, if $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{I})$ is a tableau for $\mathcal{A}$ w.r.t. $\mathcal{R}^+$, a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{A}$ can be defined similarly to the one in the proof of Lemma 14:

$$\begin{array}{rrcl} & \Delta^{\mathcal{I}} & = & \mathbf{S} \\ \text{for concept names A in } \mathit{clos}(\mathcal{A}): & A^{\mathcal{I}} & = & \{s \mid A \in \mathcal{L}(s)\} \\ \text{for individual names } a \in \mathbf{I}: & a^{\mathcal{I}} & = & \mathcal{I}(a) \in \Delta^{\mathcal{I}} \\ & R^{\mathcal{I}} & = & \begin{cases} \mathcal{E}(R)^+ & \text{if } \mathsf{Trans}(R) \\ \mathcal{E}(R) \cup \bigcup\limits_{P \sqsubseteq^* R, P \neq R} P^{\mathcal{I}} & \text{otherwise} \end{cases} \end{array}$$

where $\mathcal{E}(R)^+$ denotes the transitive closure of $\mathcal{E}(R)$. The proof that this interpretation is indeed a model of $\mathcal{A}$ w.r.t. $\mathcal{R}^+$ is identical to the one given for Lemma 14, with the additional observations that (P12) ensures that $\mathcal{I}$ satisfies assertions of the form $a\!:\!C \in \mathcal{A}$, (P13) ensures that $\mathcal{I}$ satisfies assertions of the form $(a, b)\!:\!R \in \mathcal{A}$, and (P14) ensures that $\mathcal{I}$ satisfies inequality assertions.

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

$$\begin{array}{rcl} \mathbf{S} & = & \Delta^{\mathcal{I}} \\ \mathcal{E}(R) & = & R^{\mathcal{I}} \\ \mathcal{L}(s) & = & \{C \in \mathit{clos}(\mathcal{A}) \mid s \in C^{\mathcal{I}}\} \\ \mathcal{I}(a) & = & a^{\mathcal{I}} \end{array}$$

The proof that T is a tableau for $\mathcal{A}$ is identical to the one given for Lemma 14, with the additional observations that: $\mathcal{I}$ being a model for $\mathcal{A}$ yields $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all $a\!:\!C \in \mathcal{A}$, which ensures that T satisfies (P12); analogously, T satisfies (P13) since $(a, b)\!:\!R \in \mathcal{A}$ implies $\langle a, b\rangle \in R^{\mathcal{I}}$; and finally, $a \neq b \in \mathcal{A}$ implies $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ which implies (P14).

6.2 An ABox consistency algorithm

In this section we extend the completion algorithm presented in Section 5 to decide consistency of $\mathcal{SHIQ}$ ABoxes (with respect to role hierarchies). Instead of working on completion *trees*, it works on completion *forests*, that is, a collection of completion trees whose root nodes correspond to individuals in the ABox. Moreover, the $\leqslant$-rule is modified to keep track of the identification of root nodes/individuals[5] so that we can easily build a tableau from a complete and clash-free completion forest. Finally, we modify the blocking condition so that root nodes can neither be blocked nor block any other node.

Definition 17 A *completion forest* for a $\mathcal{SHIQ}$ ABox $\mathcal{A}$ is a collection of completion trees whose root nodes are possibly connected by edges, where, besides

[5] Please recall that we do not employ the unique name assumption.

having an explicit inequality relation $\neq$, we also have an explicit equality relation $\doteq$ which is used to memorise identifications of root nodes.

The edges between root nodes are labelled with a set $\mathcal{L}(\langle x_0, y_0\rangle)$ of (possibly inverse) roles occurring in $\mathcal{A}$. Root nodes may be connected in an arbitrary way, forming a directed graph rather than a tree. However, as we do not want to introduce a special notion for edges connecting root nodes, we will abuse the existing notation be calling a root node y_0 an *R-successor* of a node x_0 if the graph has an edge $\langle x_0, y_0\rangle$ and $S \in \mathcal{L}(\langle x, y\rangle)$ for some S with $S \sqsubseteq R$. As before, y_0 an *R-neighbour* of a node x_0 if y_0 an *R-successor* of a node x_0 or x_0 an $\mathsf{Inv}(R)$*-successor* of a node y_0.

A node is *blocked* iff it is not a root node and it is either directly or indirectly blocked. A node x is *directly blocked* iff none of its ancestors are blocked, and it has ancestors x', y and y' such that

1. y is not a root node *and*
2. x is a successor of x' and y is a successor of y' *and*
3. $\mathcal{L}(x) = \mathcal{L}(y)$ and $\mathcal{L}(x') = \mathcal{L}(y')$ *and*
4. $\mathcal{L}(\langle x', x\rangle) = \mathcal{L}(\langle y', y\rangle)$.

In this case we will say that y *blocks* x.

A node y is *indirectly blocked* iff one of its ancestors is blocked, or it is a successor of a node x and $\mathcal{L}(\langle x, y\rangle) = \emptyset$; the latter condition avoids wasted expansions after an application of the $\leqslant$-rule.

Given a $\mathcal{SHIQ}$-ABox $\mathcal{A}$ and a role hierarchy $\mathcal{R}^+$, the algorithm initialises a completion forest $\mathcal{F}_\mathcal{A}$ consisting only of root nodes. More precisely, $\mathcal{F}_\mathcal{A}$ contains a root node x_0^i for each individual $a_i \in \mathbf{I}_\mathcal{A}$ occurring in $\mathcal{A}$, and an edge $\langle x_0^i, x_0^j\rangle$ for each pair of individuals (a_i, a_j) occurring in $\mathcal{F}_\mathcal{A}$ (i.e., in an assertion $(a_i, a_j) : R$ for some role R). The labels of these nodes and edges are initialised so that

$$\begin{array}{rcl} \mathcal{L}(x_0^i) & := & \{C \in \mathit{clos}(\mathcal{A}) \mid a_i{:}C \in \mathcal{A}\}, \text{ and} \\ \mathcal{L}(\langle x_0^i, x_0^j\rangle) & := & \{R \in \mathbf{R}_\mathcal{A} \mid (a_i, a_j) : R \in \mathcal{A}\}. \end{array}$$

Finally, the $\not\doteq$ relation is initialised to correspond to the inequality assertions from $\mathcal{A}$:

$$x_0^i \not\doteq x_0^j \text{ iff } a_i \neq a_j \in \mathcal{A},$$

and the $\doteq$ is initialised to be empty. $\mathcal{F}_\mathcal{A}$ is then expanded by repeatedly applying the rules from Figure 6.2.

The definitions of *clash-free* and *complete* for completion forests are the same as those given for completion trees in Definition 14, the same applies to the definition of $\sharp S^\mathcal{F}(x, C)$. If the expansion rules can be applied in such a way that they yield a complete and clash-free completion forest, then the algorithm returns "$\mathcal{A}$ is consistent w.r.t. $\mathcal{R}^+$", and "$\mathcal{A}$ and is inconsistent w.r.t. $\mathcal{R}^+$" otherwise.

Please note that the expansion rules are almost identical to those in Figure 5.3: only the $\leqslant$-rule is modified in order to deal correctly with root nodes.

For increased readability the modified rule is divided into two parts, the $\leqslant$-rule and the $\leqslant_r$-rule.

Since both the $\leqslant$-rule and the $\leqslant_r$-rule are rather complicated, they deserve some more explanation. Both rules deal with the situation where a concept $(\leqslant\ n\ R\ C) \in \mathcal{L}(x)$ requires the indentification of two R-neighbours y, z of x that contain C in their labels. Of course, y and z may only be identified if $y \neq z$ does not hold. If these conditions are met, then one of the two rules can be applied. The $\leqslant$-rule deals with the case where one of y, z is not a root-node, and this can lead to one of two possible situations, both shown in Figure 6.2.

The upper situation occurs when both y and z are successors of x. In this case, we add the label of y to that of z, and also add the label of the edge $\langle x, y\rangle$ to the label of the edge $\langle x, z\rangle$. Finally, z inherits all inequalities from y, and $\mathcal{L}(\langle x, y\rangle)$ is set to $\emptyset$, blocking y and all its successors (in order to avoid wasted expansion).

The lower situation occurs when both y and z are neighbours of x, but z is a predecessor of x. Note that, in this case, if one of y, z is a root node, then it must be z because a non-root node can never be the ancestor of a root node. Again, $\mathcal{L}(y)$ is added to $\mathcal{L}(z)$, but in this case the inverse of $\mathcal{L}(\langle x, y\rangle)$ is added to $\mathcal{L}(\langle z, x\rangle)$, because the edge $\langle x, y\rangle$ was pointing away from x while $\langle z, x\rangle$ points towards it. Again, z inherits the inequalities from y and $\mathcal{L}(\langle x, y\rangle)$ is set to $\emptyset$.

The $\leqslant_r$ rule is needed to handle the indentification of two root nodes. In this case, special care has to be taken to preserve the relations introduced into the completion forest due to role assertions in the ABox, and to memorize the identification of root nodes (this will be needed in order to construct a tableau from a complete and clash-free completion forest). The $\leqslant_r$ rule includes some additional steps that deal with these issues. Firstly, as well as adding $\mathcal{L}(y)$ to $\mathcal{L}(z)$, the relations between y and its neighbours are also added to z. Thus, for each edge $\langle y, w\rangle$ (resp. $\langle w, y\rangle$) in the forest, if $\langle z, w\rangle$ (rest. $\langle w, z\rangle$) does not already exist, then it is created and its label initialised to $\emptyset$; $\mathcal{L}(\langle y, w\rangle)$ (resp. $\mathcal{L}(\langle w, y\rangle)$) is then added to $\mathcal{L}(\langle z, w\rangle)$ (resp. $\mathcal{L}(\langle w, z\rangle)$). Secondly, y and all edges adjacent to y are removed from the forest. This will not lead to dangling trees, because all neighbours of y became neighbours of z in the previous step. Finally, the identification of y and z is recorded in the $\doteq$ relation. An example of the whole procedure is given in Figure 9.

Lemma 19 *Let $\mathcal{A}$ be a $\mathcal{SHIQ}$-ABox and $\mathcal{R}^+$ a role hierarchy. The tableaux algorithm terminates when started for $\mathcal{A}$ and $\mathcal{R}^+$.*

Proof: This proof is identical to the one of Lemma 11 with the following additional observation: In contrast to the tableaux algorithm given in the previous section, this one "removes" nodes: The $\leqslant_r$-rule removes a root node y if y must be identical to another root note due to a concept of the form $(\leqslant\ n\ S\ C)$ in a third root node. However, since no rule generates root nodes, this removal may only happen a finite number of times. Each root node is only connected to those non-root nodes that form the completion tree whose root it is. The

$\sqcap$-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 $C \in \mathcal{L}(y)$, and
	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.	$\forall S.C \in \mathcal{L}(x)$, 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	$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\forall R.C\}$
choose-rule:	if 1.	$(\bowtie\ n\ S\ C) \in \mathcal{L}(x)$, x is not indirectly blocked, and
	2.	there is an S-neighbour y of x with $\{C, \sim C\} \cap \mathcal{L}(y) = \emptyset$
	then	$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{E\}$ for some $E \in \{C, \sim C\}$
$\geqslant$-rule:	if 1.	$(\geqslant n\ S\ C) \in \mathcal{L}(x)$, x is not blocked, and
	2.	there are no n nodes $y_1, \ldots, y_n$ such that $C \in \mathcal{L}(y_i)$, y_i is an S-neighbour of x, and $y_i \neq y_j$ for $1 \leq i < j \leq n$, and
	then	create n new nodes $y_1, \ldots, y_n$ with $\mathcal{L}(\langle x, y_i\rangle) = \{S\}$, $\mathcal{L}(y_i) = \{C\}$, and $y_i \neq y_j$ for $1 \leq i < j \leq n$.
$\leqslant$-rule:	if 1.	$(\leqslant n\ S\ C) \in \mathcal{L}(x)$, x is not indirectly blocked, and
	2.	$\sharp S^{\mathcal{F}}(x, C) > n$ and there are two S-neighbours y, z of x not both root-nodes with $C \in \mathcal{L}(y), C \in \mathcal{L}(z)$, not $y \neq z$ and
	3	y is not an ancestor of z
	then	1. $\mathcal{L}(z) \longrightarrow \mathcal{L}(z) \cup \mathcal{L}(y)$ and 2. if z is an ancestor of x then $\mathcal{L}(\langle z, x\rangle) \longrightarrow \mathcal{L}(\langle z, x\rangle) \cup \mathsf{Inv}(\mathcal{L}(\langle x, y\rangle))$ else $\mathcal{L}(\langle x, z\rangle) \longrightarrow \mathcal{L}(\langle x, z\rangle) \cup \mathcal{L}(\langle x, y\rangle)$ 3. $\mathcal{L}(\langle x, y\rangle) \longrightarrow \emptyset$ 4. Set $u \neq z$ for all u with $u \neq y$

Figure 6: The Expansion Rules for $\mathcal{SHIQ}$-Aboxes

$\leqslant_r$-rule:	if 1.	$(\leqslant n\ S\ C) \in \mathcal{L}(x)$, and
	2.	$\sharp S^{\mathcal{F}}(x, C) > n$ and there are two S-neighbours y, z of x both root-nodes $C \in \mathcal{L}(y), C \in \mathcal{L}(z)$, not $y \neq z$
	then	1. $\mathcal{L}(z) \longrightarrow \mathcal{L}(z) \cup \mathcal{L}(y)$ and
		2. For all edges $\langle y, w\rangle$:
		i. if the edge $\langle z, w\rangle$ does not exist, create it with $\mathcal{L}(\langle z, w\rangle) = \emptyset$
		ii. $\mathcal{L}(\langle z, w\rangle) \longrightarrow \mathcal{L}(\langle z, w\rangle) \cup \mathcal{L}(\langle y, w\rangle)$
		3. For all edges $\langle w, y\rangle$:
		i. if the edge $\langle w, z\rangle$ does not exist, create it with $\mathcal{L}(\langle w, z\rangle) = \emptyset$
		ii. $\mathcal{L}(\langle w, z\rangle) \longrightarrow \mathcal{L}(\langle w, z\rangle) \cup \mathcal{L}(\langle w, y\rangle)$
		4. Remove y together with all edges from/to y.
		5. Set $u \neq z$ for all u with $u \neq y$.
		6. Set $y \doteq z$.

Figure 7: The $\leqslant_r$-rule that deals with root nodes.

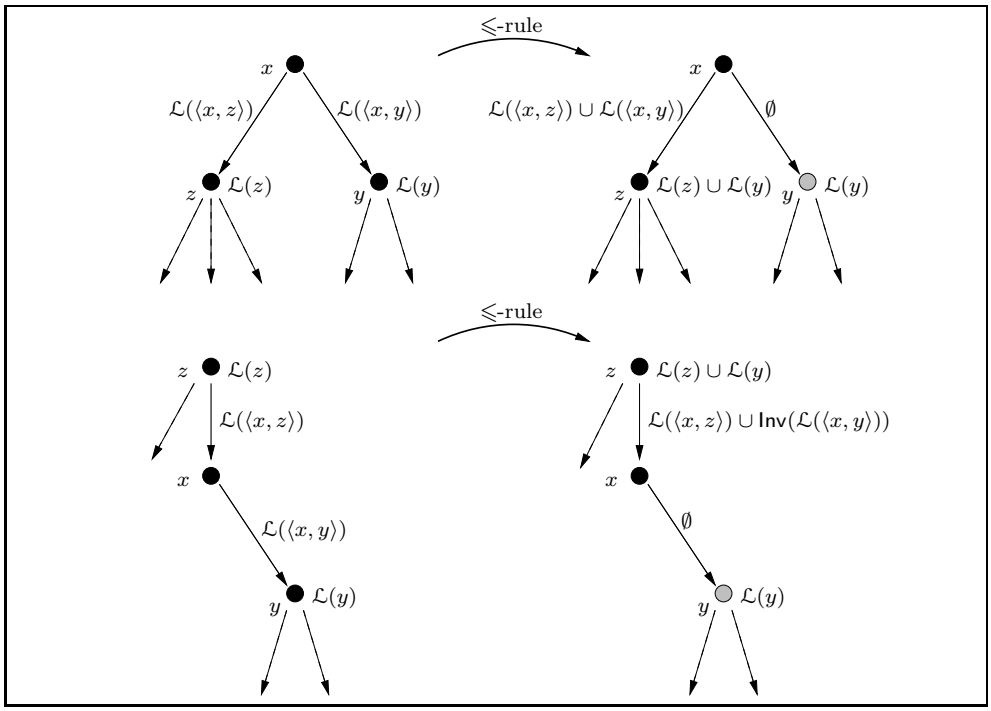

Figure 8: Effect of the $\leqslant$-rule

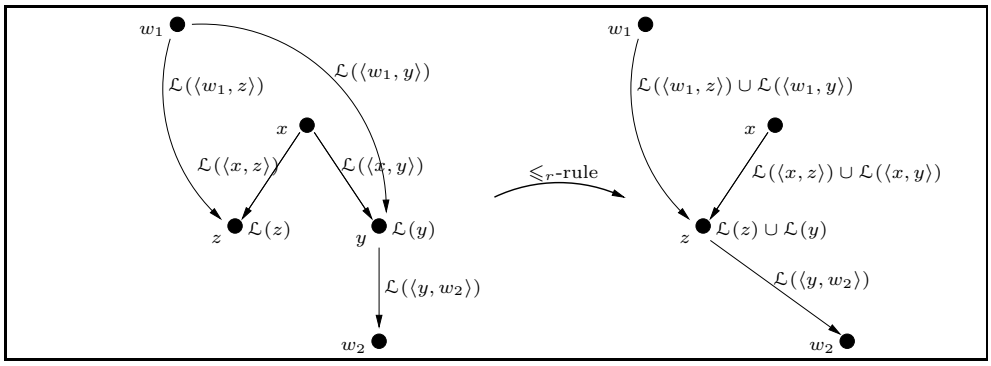

Figure 9: Effect of the $\leqslant_r$-rule

structures "hanging" on the root nodes are still completion trees, and expansion rules working on completion trees are indentical to those in the previous section. ∎

Lemma 20 *Let $\mathcal{A}$ be a $\mathcal{SHIQ}$-ABox and $\mathcal{R}^+$ a role hierarchy. If the expansion rules can be applied to $\mathcal{A}$ and R^+ such that they yield a complete and clash-free completion forest, then $\mathcal{A}$ has a tableau w.r.t. $\mathcal{R}^+$.*

Proof: Apart from the treatment of root nodes, the definition of a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{I})$ from a complete and clash-free completion forest $\mathcal{F}$ works as the one for completion trees: an individual in $\mathbf{S}$ corresponds to a *path* in $\mathcal{F}$ from some root node to some node that is not blocked, and which goes only via non-root nodes.

More precisely, let $\mathcal{F}$ be a complete and clash-free completion forest. A path is a sequence of pairs of nodes of $\mathcal{F}$ of the form $p = [(x_0, x'_0), \ldots, (x_n, x'_n)]$. For such a path we define $\mathsf{Tail}(p) := x_n$ and $\mathsf{Tail}'(p) := x'_n$. With $[p|(x_{n+1}, x'_{n+1})]$ we denote the path $[(x_0, x'_0), \ldots, (x_n, x'_n), (x_{n+1}, x'_{n+1})]$. The set $\mathsf{Paths}(\mathcal{F})$ is defined inductively as follows:

- For root nodes x_0^i of $\mathcal{F}$, $[(x_0^i, x_0^i)] \in \mathsf{Paths}(\mathcal{F})$, and
- For a path $p \in \mathsf{Paths}(\mathcal{F})$ and a node z in $\mathcal{F}$:
  - if z is a successor of $\mathsf{Tail}(p)$ and z is neither blocked nor a root node, then $[p|(z, z)] \in \mathsf{Paths}(\mathcal{F})$, or
  - if, for some node y in $\mathcal{F}$, y is a successor of $\mathsf{Tail}(p)$ and z blocks y, then $[p|(z, y)] \in \mathsf{Paths}(\mathcal{F})$.

Let us collect some simple facts that follow from the construction of $\mathsf{Paths}(\mathcal{F})$:

- Root nodes are never blocked, nor are they blocking other nodes, so paths are never extended by root-nodes. Hence, the only place where they occur in a path is in the first place.
- For each $p \in \mathsf{Paths}(\mathcal{F})$, $\mathsf{Tail}(p)$ is not blocked.
- For each $p \in \mathsf{Paths}(\mathcal{F})$, $\mathsf{Tail}(p) = \mathsf{Tail}'(p)$ iff $\mathsf{Tail}'(p)$ is not blocked.
- For each $p \in \mathsf{Paths}(\mathcal{F})$, $\mathcal{L}(\mathsf{Tail}(p)) = \mathcal{L}(\mathsf{Tail}'(p))$.

Now we can define a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{I})$ with:

$$
\begin{aligned}
\mathbf{S} &= \mathsf{Paths}(\mathcal{F}) \\
\mathcal{L}(p) &= \mathcal{L}(\mathsf{Tail}(p)) \\
\mathcal{E}(R) &= \{\langle p, [p|(x,x')]\rangle \in \mathbf{S}\times\mathbf{S} \mid x' \text{ is an } R\text{-successor of } \mathsf{Tail}(p)\} \cup \\
&\quad \{\langle [q|(x,x')], q\rangle \in \mathbf{S}\times\mathbf{S} \mid x' \text{ is an } \mathsf{Inv}(R)\text{-successor of } \mathsf{Tail}(q)\} \cup \\
&\quad \{\langle [(x_0^i, x_0^i)], [(x_0^j, x_0^j)]\rangle \in \mathbf{S}\times\mathbf{S} \mid \begin{array}{l} x_0^i, x_0^j \text{ are root nodes, and} \\ x_0^j \text{ is an } R\text{-neighbour of } x_0^i \end{array}\} \\
\mathcal{I}(a_i) &= \begin{cases} [(x_0^i, x_0^i)] & \text{if } x_0^i \text{ is a root node in } \mathcal{F} \\ [(x_0^j, x_0^j)] & \text{for } x_0^j \text{ a root node in } \mathcal{F} \text{ with } x_0^i \doteq x_0^j \end{cases}
\end{aligned}
$$

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

- T satisfies **(P1)** because $\mathcal{F}$ is clash-free.
- **(P2)** and **(P3)** are satisfied by T because $\mathcal{F}$ is complete.
- Suppose T does not satisfy **(P4)**, i.e., there are $p, q \in \mathbf{S}$ with $\forall R.C \in \mathcal{L}(p)$, $\langle p, q\rangle \in \mathcal{E}(R)$, and $C \notin \mathcal{L}(q)$. If $q = [p|(x,x')]$, then x' is an R-successor of $\mathsf{Tail}(p)$ and, due to completeness of $\mathcal{F}$, $C \in \mathcal{L}(x') = \mathcal{L}(x) = \mathcal{L}(q)$.

  If $p = [q|(x,x')]$, then x' is an $\mathsf{Inv}(R)$-successor of $\mathsf{Tail}(q)$ and, due to completeness of $\mathcal{F}$, $C \in \mathcal{L}(\mathsf{Tail}(q)) = Lab(q)$.

  If $p = [(x_0^i, x_0^i)]$ and $q = [(x_0^j, x_0^j]$ for two root nodes x_0^i, x_0^j, then x_0^j is an R-neighbour of x_0^i, and completeness of $\mathcal{F}$ yields $C \in \mathcal{L}(x_0^j)$ and hence $C \in \mathcal{L}(q)$.
- For **(P5)**, let $\exists R.C \in \mathcal{L}(p)$. Define $\mathsf{Tail}(p) = x$. Since x is not blocked, x has some R-neighbour y with $C \in \mathcal{L}(y)$.

  There are the following two possibilities:

  - y is a successor of x in $\mathcal{F}$. y can either be a root node or not.
    - Assume y is not a root node: if y is not blocked, then $q := [p|(y,y)] \in \mathbf{S}$; if y is blocked by some node z in $\mathcal{F}$, then $q := [p|(z,y)] \in \mathbf{S}$.
    - If y is a root node, then, since y is a successor of x, x is also a root node. This implies $p = [(x,x)]$ and $q = [(y,y)] \in \mathbf{S}$.

    In any of these cases, $\langle p, q\rangle \in \mathcal{E}(R)$ and $C \in \mathcal{L}(q)$ holds.

– x is an $\mathsf{Inv}(R)$-*successor* of y, then either
  * p is of the form $p = [q|(x, x')]$ with $\mathsf{Tail}(q) = y$.
  * p is of the form $p = [q|(x, x')]$ with $\mathsf{Tail}(q) = u \neq y$. x only has one predecessor in $\mathcal{F}$, hence u is not the predecessor of x. This implies $x \neq x'$, x blocks x' in $\mathcal{F}$, and u is the predecessor of x' due to the construction of Paths. Together with the definition of the blocking condition, this implies $\mathcal{L}(\langle u, x'\rangle) = \mathcal{L}(\langle y, x\rangle)$ as well as $\mathcal{L}(u) = \mathcal{L}(y)$ due to the pairwise blocking.
  * p is of the form $p = [(x, x)]$ with x being a root node. This implies that also y is a root node and $q = [(y, y)]$.

  Again, in any of these cases, $\langle p, q\rangle \in \mathcal{E}(R)$ and $C \in \mathcal{L}(q)$.

- For **(P6)**, let $\forall S.C \in \mathcal{L}(p)$, and $\langle p, q\rangle \in \mathcal{E}(R)$ for some $R \sqsubseteq^* S$ with $\mathsf{Trans}(R)$.

  If $q = [p|(x, x')]$, then x' is an R-successor of $\mathsf{Tail}(p)$ and thus $\forall R.C \in \mathcal{L}(x')$ because otherwise the $\forall_+$-rule would be applicable. From $\mathcal{L}(q) = \mathcal{L}(x) = \mathcal{L}(x')$ it follows that $\forall R.C \in \mathcal{L}(q)$.

  If $p = [q|(x, x')]$, then x' is an $\mathsf{Inv}(R)$-successor of $\mathsf{Tail}(q)$ and, due to completeness of $\mathcal{F}$, $\forall S.C \in \mathcal{L}(\mathsf{Tail}(q)) = Lab(q)$.

  If $p = [(x_0^i, x_0^i)]$ and $q = [(x_0^j, x_0^j)]$ for two root nodes x_0^i, x_0^j, then x_0^j is an R-neighbour of x_0^i, and completeness of $\mathcal{F}$ yields $\forall R.C \in \mathcal{L}(x_0^j) = \mathcal{L}(q)$.

- **(P7)** holds because of the symmetric definition of the mapping $\mathcal{E}$.

- **(P8)** is due to the definition of (R-)neighbours and (R-)successor: If a node y is an R-neighbour (resp. R-successor) of a node x and $R \sqsubseteq^* S$, then y is also S-neighbour (resp. S-successor) of x.

- Suppose **(P9)** were not satisfied. Hence there is some $p \in \mathbf{S}$ with $(\leqslant n\ S\ C) \in \mathcal{L}(p)$ and $\sharp S^T(p, C) > n$. We will show that this implies $\sharp S^{\mathcal{F}}(\mathsf{Tail}(p), C) > n$ which is a contradiction to either the clash-freeness or completeness of $\mathcal{F}$.

  Define $x := \mathsf{Tail}(p)$ and $P := S^T(p, C)$. Due to the assumption, we have $\sharp P > n$. Like in the proof of Lemma 16, we distinguish two cases:

  – P contains only paths of the form $[p|(y, y')]$ and $[(x_0^{i_\ell}, x_0^{i_\ell})]$. We claim that the function Tail' is injective on P. If we assume that there are two paths $q_1, q_2 \in P$ with $q_1 \neq q_2$ and $\mathsf{Tail}'(q_1) = \mathsf{Tail}'(q_2) = y'$ then this implies that each q_i is of the form $q_i = [p|(y_i, y')]$ or $q_i = [(y', y')]$. From $q_1 \neq q_2$, we have that $q_i = [p|(y_i, y')]$ holds for $i \in \{1, 2\}$. Since root nodes occur only in the beginning of paths, $q_i = [p|(y_i, y')]$ and $q_j = [(y', y')]$ is not possible. Hence $q_i = [p|(y_i, y')]$ for both $i \in \{1, 2\}$. If y' is not blocked in $\mathcal{F}$, then $y_1 = y' = y_2$ holds, contradicting $q_1 \neq q_2$. If y' is blocked in $\mathcal{F}$, then both y_1 and y_2 block y', which implies $y_1 = y_2$, again a contradiction.

Since Tail' is injective on P, it holds that $\sharp P = \sharp\,\mathsf{Tail}'(P)$. Moreover, for each $y' \in \mathsf{Tail}'(P)$, y' is an S-successor of x and $C \in \mathcal{L}(y')$. This implies $\sharp S^{\mathcal{F}}(x, C) > n$.

– P contains a path q where p is of the form $p = [q|(x, x')]$. Hence p is not of the form $[(x_0^i, x_0^i)]$, and this case is identical to the one in Lemma 16.

- For **(P10)**, assume $(\geqslant n\ S\ C) \in \mathcal{L}(p)$. This implies that there exist n individuals $y_1, \ldots, y_n$ in $\mathcal{F}$ such that each y_i is an S-neighbour of $x = \mathsf{Tail}(p)$ and $C \in \mathcal{L}(y_i)$. We claim that, for each of these individuals, there is a path q_i such that $\langle p, q_i\rangle \in \mathcal{E}(S)$, $C \in \mathcal{L}(q_i)$, and $q_i \neq q_j$ for all $1 \leq i < j \leq n$. Obviously, this implies $\sharp S^{\mathcal{T}}(p, C) \geqslant n$. For each y_i there are three possibilities:

  – y_i is an S-successor of x and y_i is not blocked in $\mathcal{F}$. Then $q_i = [p|(y_i, y_i)]$ or y_i is a root node and $q_i = [(y_i, y_i)]$, and q_i is a path with the desired properties.

  – y_i is an S-successor of x and y_i is blocked in $\mathcal{F}$ by some node z. Then $q_i = [p|(z, y_i)]$ is the path with the desired properties. Since the same z may block several of the y_js, it is indeed necessary to include y_i explicitly into the path to make them distinct.

  – x is an $\mathsf{Inv}(S)$-successor of y_i. There may be at most one such y_i if x is not a root node. Hence either p is of the form $p = [q|(x, x')]$ with $\mathsf{Tail}(q) = y_i$, or $p = [(x, x)]$ and $q_i = [(y_i, y_i)]$. Again, q_i has the desired properties and, obviously, q is distinct from all other paths q_j.

- For **(P11)**, let $(\bowtie\ \ n\ S\ C) \in \mathcal{L}(p)$ and $\langle p, q\rangle \in \mathcal{E}(S)$. If $q = [p|(x, x')]$ or $q = [(x', x')]$, then x' is an S-successor of $\mathsf{Tail}(p)$ and thus $\{C, \sim C\} \cap \mathcal{L}(x') \neq \emptyset$ must hold since the *choose*-rule is not applicable. Since $\mathcal{L}(q) = \mathcal{L}(x) = \mathcal{L}(x')$ we have $\{C, \sim C\} \cap \mathcal{L}(q) \neq \emptyset$. If $p = [q|(x, x')]$, then x' is an $\mathsf{Inv}(S)$-successor of $\mathsf{Tail}(q)$ and thus $\{C, \sim C\} \cap \mathcal{L}(\mathsf{Tail}(q)) \neq \emptyset$ must hold (since x' is not indirectly blocked and the *choose*-rule is not applicable), hence $\{C, \sim C\} \cap \mathcal{L}(q) \neq \emptyset$.

- **(P12)** is due to the fact that, when the tableaux algorithm is started for an ABox $\mathcal{A}$, the initial completion forest $\mathcal{F}_{\mathcal{A}}$ contains, for each individual name a_i occurring in $\mathcal{A}$, a root node x_0^i with

$$\mathcal{L}(x_0^i) = \{C \in \mathit{clos}(\mathcal{A}) \mid a_i{:}C \in \mathcal{A}\}.$$

  The tableaux algorithm never blocks root individuals, and, for each root node x_0^i that is removed by the $\leqslant$-rule, there is another root node x_0^j with $x_0^i \doteq x_0^j$. For $\mathcal{I}(x_0^i) = x_0^j$, we can say that x_0^j has "inherited" all concepts in the label of x_0^i, i.e. $\{C \in \mathit{clos}(\mathcal{A}) \mid a_i{:}C \in \mathcal{A}\} \in \mathcal{L}(x_0^j)$. Hence for each individual name a_i that occurs in $\mathcal{A}$, there is a path $[(x_0^j, x_0^j)] \in \mathbf{S}$ with $C \in \mathcal{L}([(x_0^k, x_0^k)])$ for each $a_i{:}C \in \mathcal{A}$.

- **(P13)** is satisfied for the same reasons as **(P12)**, i.e. root nodes are never blocked, and if a root node is removed, there is another root node that "inherits" all its role-relationships.

- **(P14)** is satisfied because the $\leqslant_r$-rule can not identify two root nodes x_0^i, y_0^i for which $x_0^i \neq y_0^i$ holds. The relation $\neq$ is initialised with all the inequality assertions from $\mathcal{A}$. ∎

Lemma 21 *Let $\mathcal{A}$ be a $\mathcal{SHIQ}$-ABox and $\mathcal{R}^+$ a role hierarchy. If $\mathcal{A}$ has a tableau w.r.t. $\mathcal{R}^+$., then the expansion rules can be applied to $\mathcal{A}$ and $\mathcal{R}^+$ such that they yield a complete and clash-free completion forests.*

Proof: Let $T = (\mathbf{S}, \mathcal{L}, \mathcal{E}, \mathcal{I})$ be a tableau for $\mathcal{A}$ and $\mathcal{R}^+$. Similarly to the proof of Lemma 17, we can use T to trigger the application of the expansion rules such that they yield a completion forest $\mathcal{F}$ that is both complete and clash-free. Again, we use a function π which maps the nodes of $\mathcal{F}$ to elements of $\mathbf{S}$, and we steer the application of the $\sqcup$-rule, the $\leqslant$-rule, and the $\leqslant_r$-rule such that $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ holds for all nodes x of the completion forest.

Since T is a tableau for $\mathcal{A}$, for each assertion $a_i : C \in \mathcal{A}$, we have $C \in \mathcal{L}(\mathcal{I}(a_i))$. When applying the expansion rules, the application of the non-deterministic $\sqcup$-rule is driven by the labelling $\mathcal{L}$ in the tableau T. To this purpose, we define a mapping π which maps the nodes of $\mathcal{F}$ to elements of $\mathbf{S}$, and we steer the application of the $\sqcup$-rule such that

$$\left.\begin{array}{l} \mathcal{L}(x) \subseteq \mathcal{L}(\pi(x)) \\ \text{if } y \text{ is an } S\text{-neighbour of } x \text{ then } \langle \pi(x), \pi(y) \rangle \in \mathcal{E}(S) \\ x \neq y \text{ implies } \pi(x) \neq \pi(y) \end{array}\right\} (*)$$

Claim: Let $\mathcal{F}$ be a completion forest and π a function that satisfies $(*)$. If a rule is applicable to $\mathcal{F}$ then the rule is applicable to $\mathcal{F}$ in a way that yields a completion forest $\mathcal{F}$ and a function π that satisfy $(*)$. This claim is now proved by induction, similar to the proof of Lemma 17:

- If the $\sqcap$-rule can be applied to x in $\mathcal{F}$ with $C_1 \sqcap C_2 \in \mathcal{L}(x)$, 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 $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$.

- If the $\sqcup'$-rule can be applied to x in $\mathcal{F}$ with $C_1 \sqcup C_2 \in \mathcal{L}(x)$, then $C \in \{C_1, C_2\}$ is in $\mathcal{L}(\pi(x))$, and C_1 or C_2 is added to $\mathcal{L}(x)$ by the $\sqcup$-rule. Hence the $\sqcup$-rule can be applied such that, after its application, we have $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ for all nodes x in $\mathcal{F}$.

- If the $\exists$-rule can be applied to x in $\mathcal{F}$ 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 we thus can define $\pi(y) := t$ for some t with $C_1 \in \mathcal{L}(t)$ and $\langle \pi(x), t \rangle \in \mathcal{E}(R)$. Hence we have $\mathcal{L}(y) = \{C_1\} \subseteq \mathcal{L}(\pi(y))$ and $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(R)$ for the new R-successor y of x.

- If the $\forall$-rule can be applied to x in $\mathcal{F}$ 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, due to (P4), $C_1 \in \mathcal{L}(\pi(y))$. The $\forall$-rule adds C_1 to $\mathcal{L}(y)$ and thus preserves $\mathcal{L}(y) \subseteq \mathcal{L}(\pi(y))$.

- If the $\forall_+$-rule can be applied to x in $\mathcal{F}$ with $C = \forall S.C_1 \in \mathcal{L}(x)$, $\mathsf{Trans}(R)$, $R \sqsubseteq S$, and y an R-neighbour of x, then $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(R)$, and thus, due to (P6), $\forall R.C_1 \in \mathcal{L}(\pi(y))$. The $\forall_+$-rule adds $\forall R.C_1$ to $\mathcal{L}(y)$ and thus preserves $\mathcal{L}(y) \subseteq \mathcal{L}(\pi(y))$.

- If the $\geqslant$-rule can be applied to x in $\mathcal{F}$ with $C = (\geqslant n\ S\ C)_1 \in \mathcal{L}(x)$, then $C \in \mathcal{L}(\pi(x))$ and, by (P10), there are $t_1, \ldots, t_n \in \mathbf{S}$ with $\langle \pi(x), t_i \rangle \in \mathcal{E}S$ and $C_1 \in \mathcal{L}(t_i)$. Hence applying the $\geqslant$-rule and extending π by $\pi(y_i) = t_i$ yields a completion forest and a mapping π that still satisfies $(*)$.

- If the *choose*-rule can be applied to x with $(\bowtie\ n\ S\ C) \in \mathcal{L}(x)$ and an S-neighbour y of x, then $(\bowtie\ n\ S\ C) \in \mathcal{L}(\pi(x))$ and $\langle x, y \rangle \in \mathcal{E}(S)$ and, due to (P11) either $C \in \mathcal{L}(y)$ or $\sim C \in \mathcal{L}(y)$. Hence the *choose*-rule can be applied to x in such a way that $(*)$ is satisfied.

- Let the $\leqslant$- or the $\leqslant_r$-rule be applicable to x with $C = (\leqslant n\ S\ C_1) \in \mathcal{L}(x)$ and more than n S-neighbours. Since $(\leqslant\ n\ S\ C_1) \in \mathcal{L}(\pi(x))$ and T satisfies (P9), there are at most n elements y_j with $\langle \pi(x), y_j \rangle \in \mathcal{E}(S)$ and thus, due to $(*)$, there are two S-neighbours t_1, t_2 of x with $\pi(t_1) = \pi(t_2)$.

  - In the case that one of the t_i is not a root node, the $\leqslant$-rule is applicable. $\pi(y) = \pi(z)$ implies that $y \neq z$ cannot hold because of $(*)$, and y, z can be chosen such that y is not an ancestor of z. Hence the $\leqslant$-rule can be applied without violating $(*)$.
  - If both t_1, t_2 are root nodes, then $(*)$ implies $\mathcal{L}(t_1) \cup \mathcal{L}(t_2) \subseteq \mathcal{L}(\pi(t_1))$, hence setting $\mathcal{L}(t_1) = \mathcal{L}(t_1) \cup \mathcal{L}(t_2)$ preserves this subset-relation. Moreover, together with $(*)$, we have that, if z is an S-neighbour of t_1 or of t_2, then $\langle \pi(z), \pi(t_1) \rangle \in \mathcal{E}(S)$. Hence, adding edges $\langle z, w \rangle$ and $\langle w, z \rangle$ as necessary, then setting $\mathcal{L}(\langle z, t_1 \rangle) = \mathcal{L}(\langle z, t_1 \rangle) \cup \mathcal{L}(\langle z, t_2 \rangle)$ and $\mathcal{L}(\langle t_1, z \rangle) = \mathcal{L}(\langle t_1, z \rangle) \cup \mathcal{L}(\langle t_2, z \rangle)$, yields a forest that satisfies $(*)$. Summing up, the $\leqslant_r$-rule can be applied in such a way that it yields a completion forest and a mapping π that satisfy $(*)$.

This claim proves the lemma as follows. By setting $\pi(x_0^i) = \mathcal{I}(a_i)$ for each root node x_0^i we obtain a mapping π that satisfies $(*)$ for the initial completion forest $\mathcal{F}_\mathcal{A}$ due to (P12), (P13), and (P14). From Lemma 19, the tableau-construction triggered by $\mathcal{F}_\mathcal{A}$ terminates with a complete forest $\mathcal{F}$. Moreover, whenever a rule is applicable, then it can be applied in a way that maintains $(*)$. Hence, there is a complete forest $\mathcal{F}$ and a mapping π that satisfies $(*)$. Yet, this implies that $\mathcal{F}$ must be clash free:

- $\mathcal{F}$ cannot contain a node x such that $\{C, \neg C\} \in \mathcal{L}(x)$ because $\mathcal{L}(x) \subseteq \mathcal{L}(\pi(x))$ and hence (P1) would be violated for $\pi(x)$.

- $\mathcal{F}$ cannot contain a node x with $(\leqslant n\ S\ C) \in \mathcal{L}(x)$ and $n+1$ S-neighbours $y_0, \ldots y_n$ of x with $C \in \mathcal{L}(y_i)$ and $y_i \neq y_j$ for $0 \leq i < j \leq n$, because $(\leqslant n\ S\ C) \in \mathcal{L}(\pi(x))$, and, since $y_i \neq y_j$ implies $\pi(y_i) \neq \pi(y_j)$, $\sharp S^{\mathcal{T}}(\pi(x), C) > n$ would hold which contradicts (P9). ∎

6.3 ABox Reasoning with Respect to a Terminology

Using the same techniques and arguments as in Section 3.1, general concept inclusion axioms can be internalised. However, as it is not guaranteed that all root nodes will be connected, it is necessary to add the concept $C_{\mathcal{T}} \sqcap \forall U.C_{\mathcal{T}}$ to the label of every root node when the completion forest is initialised. This is equivalent to adding an assertion $a : (C_{\mathcal{T}} \sqcap \forall U.C_{\mathcal{T}})$ to the ABox for every individual a occurring in it. We thus have the following theorem:

Theorem 6 *The tableaux algorithm is a decision procedure for the consistency of $\mathcal{SHIQ}$-ABoxes with respect to role hierarchies and terminologies.*

References

[Baa90] Franz Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. Technical Report RR-90-13, DFKI, Kaiserslautern, Deutschland, 1990. An abridged version appeared in *Proc. of IJCAI-91*, pp. 446–451.

[BBH96] F. Baader, M. Buchheit, and B. Hollunder. Cardinality restrictions on concepts. *Artificial Intelligence*, 88(1–2):195–213, 1996.

[BBN+93] 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.

[DL96] G. De Giacomo and M. Lenzerini. Tbox and Abox reasoning in expressive description logics. In *Proc. of KR-96*, pages 316–327. M. Kaufmann, Los Altos, 1996.

[DM98] G. De Giacomo and F. Massacci. Combining deduction and model checking into tableaux and algorithms for converse-pdl. *Information and Computation*, 1998. To appear.

[HB91] B. Hollunder and F. Baader. Qualifying number restrictions in concept languages. In *Proc. of KR-91*, pages 335–346, Boston, MA, USA, 1991.

[HG97] 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, *Proc. of DL'97*, pages 25–28, 1997.

[HNS90] B. Hollunder, W. Nutt, and M. Schmidt-Schauss. Subsumption algorithms for concept description languages. In *ECAI-90*, Pitman Publishing, London, 1990.

[HS98] I. Horrocks and U. Sattler. A description logic with transitive and inverse roles and role hierarchies. Technical Report 98-05, LuFg Theoretical Computer Science, RWTH Aachen, 1998. Available via www: `http://www-lti.informatik.rwth-aachen.de/Forschung/Papers.html`.

[HS99] I. Horrocks and U. Sattler. A description logic with transitive and inverse roles and role hierarchies. *J. of Logic and Computation*, 1999. To appear.

[HST98] I. Horrocks, U. Sattler, and S. Tobies. A PSpace-algorithm for deciding $\mathcal{ALCNI}_{R^+}$-satisfiability. LTCS-Report 98-08, LuFg Theoretical Computer Science, RWTH Aachen, Germany, 1998.

[Sat96] 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 *LNAI*. Springer-Verlag, 1996.

[Sat98] U. Sattler. *Terminological knowledge representation systems in a process engineering application*. PhD thesis, RWTH Aachen, 1998.

[Sch91] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of IJCAI-91*, pages 466–471, Sydney, 1991.

[SS88] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with unions and complements. Technical Report SR-88-21, FB Informatik, Univ. Kaiserslautern, Kaiserslautern, Deutschland, 1988.

[Tob99] S. Tobies. A PSpace algorithm for graded modal logic. In H. Ganzinger, editor, *Automated Deduction – CADE-16, 16th International Conference on Automated Deduction*, LNAI 1632, pages 52–66, Trento, Italy, July 7–10, 1999. Springer-Verlag.