The Even More Irresistible $\mathcal{SROIQ}$

Ian Horrocks, Oliver Kutz, and Ulrike Sattler

School of Computer Science, The University of Manchester,
Kilburn Building, Oxford Road, Manchester, M13 9PL, UK.
{Horrocks, Kutz, Sattler}@cs.man.ac.uk

Abstract

We describe an extension of the description logic underlying OWL-DL, $\mathcal{SHOIN}$, with a number of expressive means that we believe will make it more useful in practise. Roughly speaking, we extend $\mathcal{SHOIN}$ with all expressive means that were suggested to us by ontology developers as useful additions to OWL-DL, and which, additionally, do not affect its decidability. We consider complex role inclusion axioms of the form $R \circ S \dot{\sqsubseteq} R$ or $S \circ R \dot{\sqsubseteq} R$ to express propagation of one property along another one, which have proven useful in medical terminologies. Furthermore, we extend $\mathcal{SHOIN}$ with reflexive, symmetric, transitive, and irreflexive roles, disjoint roles, a universal role, and constructs $\exists R.\mathsf{Self}$, allowing, for instance, the definition of concepts such as a "narcist". Finally, we consider negated role assertions in Aboxes and qualified number restrictions. The resulting logic is called $\mathcal{SROIQ}$.

We present a rather elegant tableau-based reasoning algorithm: it combines the use of automata to keep track of universal value restrictions with the techniques developed for $\mathcal{SHOIQ}$. We believe that $\mathcal{SROIQ}$ could serve as a logical basis for possible future extensions of OWL-DL.

Keywords: description logics; KR languages; ontology methodology.

1 Introduction

We describe an extension, called $\mathcal{SROIQ}$, of the description logic (DL) $\mathcal{SHOIN}$ (14) underlying OWL-DL (9).[1] $\mathcal{SHOIN}$ can be said to provide most expressive means that one could reasonably expect from the logical basis of an ontology language, and to constitute a good compromise between expressive power and computational complexity/practicability of reasoning. However, it lacks e.g. qualified number restrictions which are present in the DL considered here since they are required in various applications (21) and do not pose problems (13). That is, we extend $\mathcal{SHOIQ}$—which is $\mathcal{SHOIN}$ with qualified number restrictions—and extend the work begun in (8).

Since OWL-DL is becoming more widely used, it turns out that it lacks a number of expressive means which—when considered carefully—can be added without causing too much difficulties for automated reasoning. We will extend $\mathcal{SHOIQ}$ with these expressive means and, although they are not completely independent in that some of them can be expressed using others, first present them together with some examples. Recall that, in $\mathcal{SHOIQ}$, we can already state that a role is transitive or the subrole or the inverse of another one. In addition, $\mathcal{SROIQ}$ allows for the following:

1. *disjoint roles*. Most DLs can be said to be "unbalanced" since they allow to express disjointness on concepts but not on roles, despite the fact that role disjointness is quite natural and can generate new subsumptions or inconsistencies in the presence of role hierarchies and number restrictions. E.g., the roles `sister` and `mother` or `partOf` and `hasPart` should be declared as being disjoint.

2. *reflexive and irreflexive roles*. These features are of minor interest when considering TBoxes only, yet they add some useful constraints on ABoxes, especially in the presence of number restrictions. E.g., the role `knows` should be declared as being reflexive, and the role `hasSibling` should be declared as being irreflexive.

3. *negated role assertions*. Most Abox formalisms only allow for positive role assertions (with few exceptions (1; 5)), whereas $\mathcal{SROIQ}$ also allows for statements such as $(\texttt{John}, \texttt{Mary}) : \neg\texttt{likes}$. In the presence of

[1] OWL also includes *datatypes*, a simple form of *concrete domain* (4). These can, however, be treated exactly as in $\mathcal{SHOQ}(\mathbf{D})/\mathcal{SHOQ}(\mathbf{D}_n)$ (10; 18), so we will not complicate our presentation by considering them here.

complex role inclusions, negated role assertions can be quite useful and, like disjoint roles, they overcome a certain asymmetry in expressivity.

4. $\mathcal{SROIQ}$ provides complex role inclusion axioms of the form $R \circ S \dot{\sqsubseteq} R$ and $S \circ R \dot{\sqsubseteq} R$ that were first introduced in $\mathcal{RIQ}$ (12). For example, w.r.t. the axiom $\mathtt{owns} \circ \mathtt{hasPart} \dot{\sqsubseteq} \mathtt{owns}$, and the fact that each car contains an engine $\mathtt{Car} \dot{\sqsubseteq} \exists \mathtt{hasPart}.\mathtt{Engine}$, an owner of a car is also an owner of an engine, i.e., the following subsumption is implied: $\exists \mathtt{owns}.\mathtt{Car} \sqsubseteq \exists \mathtt{owns}.\mathtt{Engine}$.

5. $\mathcal{SROIQ}$ provides the *universal role* U. Together with nominals (which are also provided by $\mathcal{SHOIQ}$), this role is a prominent feature of hybrid logics (6). Nominals can also be viewed as a powerful generalisation of *ABox individuals* (19; 10). They occur naturally in ontologies, e.g., when describing a class such as $\mathsf{EUCountries}$ by enumerating its members.

6. Finally, $\mathcal{SROIQ}$ allows for concepts of the form $\exists R.\mathsf{Self}$ which can be used to express "local reflexivity" of a role R, e.g., to define the concept "narcist" as $\exists \mathtt{likes}.\mathsf{Self}$.

Besides a Tbox and an Abox, $\mathcal{SROIQ}$ provides a so-called *Rbox* to gather all statements concerning roles.

$\mathcal{SROIQ}$ is designed to be of similar practicability as $\mathcal{SHIQ}$. The tableau algorithm for $\mathcal{SROIQ}$ presented here is essentially a combination of the algorithms for $\mathcal{RIQ}$ and $\mathcal{SHOIQ}$. Even though the additional expressive means require certain adjustments, these adjustments do not add new sources of non-determinism, and, subject to empirical verification, are believed to be "harmless" in the sense of not significantly degrading typical performance as compared with the $\mathcal{SHOIQ}$ algorithm.

More precisely, we employ the same technique using finite automata as in (12) to handle role inclusions $R \circ S \dot{\sqsubseteq} R$ and $S \circ R \dot{\sqsubseteq} R$. This involves a pre-processing step which takes an Rbox and builds, for each role R, a finite automaton that accepts exactly those words $R_1 \ldots R_n$ such that, in each model of the Rbox, $\langle x, y \rangle \in (R_1 \ldots R_n)^{\mathcal{I}}$ implies $\langle x, y \rangle \in R^{\mathcal{I}}$. These automata are then used in the tableau expansion rules to check, for a node x with $\forall R.C \in \mathcal{L}(x)$ and an $R_1 \ldots R_n$-neighbour y of x, whether to add C to $\mathcal{L}(y)$. Even though the pre-processing step might appear a little cumbersome, the usage of the automata in the algorithm makes it quite elegant and compact.

Moreover, the algorithm for $\mathcal{SROIQ}$ has, similar to the one for $\mathcal{SHOIQ}$, excellent “pay as you go” characteristics. For instance, in case only expressive means of $\mathcal{SHIQ}$ are used, the new algorithm will behave just like the algorithm for $\mathcal{SHIQ}$.

We believe that the combination of properties described above makes $\mathcal{SROIQ}$ a very useful basis for future extensions of OWL DL.

2 The Logic $\mathcal{SROIQ}$

In this section, we introduce the DL $\mathcal{SROIQ}$. This includes the definition of syntax, semantics, and inference problems.

2.1 Roles, Role Hierarchies, and Role Assertions

Definition 1 *Let* $\mathbf{C}$ *be a set of* ***concept names*** *including a subset* $\mathbf{N}$ *of* ***nominals****,* $\mathbf{R}$ *a set of* ***role names*** *including the universal role* U*, and* $\mathbf{I} = \{a, b, c \ldots\}$ *a set of* ***individual names****. The set of* ***roles*** *is* $\mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$*, where a role* R^- *is called the* ***inverse role*** *of* R*.*

As usual, 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* ***valuation*** $\cdot^\mathcal{I}$ *which associates, with each role name* R*, a binary relation* $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$*, with the universal role* U *the universal relation* $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$*, with each concept name* C *a subset* $C^\mathcal{I} \subseteq \Delta^\mathcal{I}$*, where* $C^\mathcal{I}$ *is a singleton subset if* $C \in \mathbf{N}$*, and with each individual name* a *an element* $a^\mathcal{I} \in \Delta^\mathcal{I}$*. Inverse roles are interpreted as usual, i.e., for each role* $R \in \mathbf{R}$*, we have*

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

Obviously, $(U^-)^\mathcal{I} = (U)^\mathcal{I}$. Note that, unlike in the cases of $\mathcal{SHIQ}$ or $\mathcal{SHOIQ}$, we did not introduce *transitive role names*. This is because, as will become apparent below, role box assertions can be used to force roles to be transitive.

To avoid considering roles such as R^{--}, we define a function Inv on roles such that $\mathsf{Inv}(R) = R^-$ if $R \in \mathbf{R}$ is a role name, and $\mathsf{Inv}(R) = S \in \mathbf{R}$ if $R = S^-$.

Since we will often work with a string of roles, it is convenient to extend both $\cdot^\mathcal{I}$ and $\mathsf{Inv}(\cdot)$ to such strings: if $w = R_1 \ldots R_n$ for R_i roles, then we set $w^\mathcal{I} = R_1^\mathcal{I} \circ \ldots \circ R_n^\mathcal{I}$ and $\mathsf{Inv}(w) = \mathsf{Inv}(R_n) \ldots \mathsf{Inv}(R_1)$, where $\circ$ denotes composition of binary relations.

A *role box* $\mathcal{R}$ consists of two components. The first component is a *role hierarchy* $\mathcal{R}_h$ which consists of (generalised) *role inclusion axioms*. Typically, these statements are of the form $R \dot\sqsubseteq S$, $RS \dot\sqsubseteq S$, and $SR \dot\sqsubseteq S$. However, we also allow role inclusion axioms of the form $S^- \dot\sqsubseteq S$, $SS \dot\sqsubseteq S$, and $w \dot\sqsubseteq S$, where w is a *finite string* of roles of a certain shape, for details see below.

The second component is a set $\mathcal{R}_a$ of *role assertions* stating, for instance, that a role R must be interpreted as an irreflexive relation, or that two (possibly inverse) roles R and S are to be interpreted as *disjoint* binary relations.

We start with the definition of a (regular) role hierarchy, whose definition involves a certain ordering on roles, called *regular*. A strict partial order $\prec$ on a set A is an irreflexive and transitive relation on A. A strict partial order $\prec$ on the set of roles $R \cup \{R^- \mid R \in \mathbf{R}\}$ is called a **regular order**, if $\prec$ satisfies additionally

$$S \prec R \iff S^- \prec R,$$

for all roles R and S. Note, in particular, that the irreflexivity of $\prec$ ensures that neither $S^- \prec S$ nor $S \prec S^-$ hold.

Definition 2 ((Regular) Role Inclusion Axioms) *Let $\prec$ be a regular order on roles. A* ***role inclusion axiom*** *(RIA for short) is an expression of the form $w \dot\sqsubseteq R$, where w is a finite string of roles, and R is a role name. A* ***role hierarchy*** *$\mathcal{R}_h$, then, is a finite set of RIAs.*

An interpretation $\mathcal{I}$ ***satisfies*** *a role inclusion axiom $S_1 \ldots S_n \dot\sqsubseteq R$, if*

$$S_1^\mathcal{I} \circ \ldots \circ S_n^\mathcal{I} \subseteq R^\mathcal{I},$$

where $\circ$ stands for the composition of binary relations. An interpretation is a ***model*** *of a role hierarchy R_h, if it satisfies all RIAs in R_h, written $\mathcal{I} \models R_h$. A RIA $w \dot\sqsubseteq R$ is* ***$\prec$-regular*** *if R is a role name, and*

1. *$w = RR$, or*

2. *$w = R^-$, or*

3. *$w = S_1 \ldots S_n$ and $S_i \prec R$, for all $1 \leq i \leq n$, or*

4. *$w = RS_1 \ldots S_n$ and $S_i \prec R$, for all $1 \leq i \leq n$, or*

5. $w = S_1 \dots S_n R$ and $S_i \prec R$, for all $1 \leq i \leq n$.

*Finally, a role hierarchy $\mathcal{R}_h$ is said to be **regular** if there exists a regular order $\prec$ on roles such that each RIA in $\mathcal{R}_h$ is $\prec$-regular.*

Regularity prevents a role hierarchy from containing cyclic dependencies. For instance, the role hierarchy

$$\{RS \mathrel{\dot\sqsubseteq} S, \quad RT \mathrel{\dot\sqsubseteq} R, \quad VT \mathrel{\dot\sqsubseteq} T, \quad VS \mathrel{\dot\sqsubseteq} V\}$$

is not regular because it would require $\prec$ to satisfy $S \prec V \prec T \prec R \prec S$, which would imply $S \prec S$, thus contradicting irreflexivity. Such cyclic dependencies are known to lead to undecidability (12).

Also, note that RIAs of the form $RR^- \mathrel{\dot\sqsubseteq} R$, which would imply (a weak form of) reflexivity of R, are not regular according to the definition of regular orderings. However, an equivalent condition on R can be imposed by using the concept $\exists R.\mathsf{Self}$; see below.

From the definition of the semantics of inverse roles, it follows immediately that

$$\langle x, y\rangle \in w^{\mathcal{I}} \text{ iff } \langle y, x\rangle \in \mathsf{Inv}(w)^{\mathcal{I}}.$$

Hence, each model satisfying $w \mathrel{\dot\sqsubseteq} S$ also satisfies $\mathsf{Inv}(w) \mathrel{\dot\sqsubseteq} \mathsf{Inv}(S)$ (and vice versa), and thus the restriction to those RIAs with role *names* on their right hand side does not have any effect on expressivity.

Given a role hierarchy $\mathcal{R}_h$, we define the relation $\mathrel{\underline{\circledast}}$ to be the transitive-reflexive closure of $\mathrel{\dot\sqsubseteq}$ over $\{R \mathrel{\dot\sqsubseteq} S, \mathsf{Inv}(R) \mathrel{\dot\sqsubseteq} \mathsf{Inv}(S) \mid R \mathrel{\dot\sqsubseteq} S \in \mathcal{R}_h\}$. A role R is called a **sub-role** (resp. **super-role**) of a role S if $R \mathrel{\underline{\circledast}} S$ (resp. $S \mathrel{\underline{\circledast}} R$). Two roles R and S are **equivalent** ($R \equiv S$) if $R \mathrel{\underline{\circledast}} S$ and $S \mathrel{\underline{\circledast}} R$.

Note that, due to restriction (3) in the definition of $\prec$-regularity, we also restrict $\mathrel{\underline{\circledast}}$ to be acyclic, and thus regular role hierarchies never contain two equivalent roles.[2]

Next, let us turn to the second component of Rboxes, the role assertions. For an interpretation $\mathcal{I}$, we define $Diag^{\mathcal{I}}$ to be the set $\{\langle x, x\rangle \mid x \in \Delta^{\mathcal{I}}\}$. Note that, since the interpretation is fixed in any given model, we disallow the universal role to appear in role assertions.

[2]This is not a serious restriction for, if $\mathcal{R}$ contains $\mathrel{\underline{\circledast}}$ cycles, we can simply choose one role R from each cycle and replace all other roles in this cycle with R in the input Rbox, Tbox and Abox (see below).

Definition 3 (Role Assertions) *For roles $R, S \neq U$, we call the assertions* $\mathsf{Ref}(R)$*,* $\mathsf{Irr}(R)$*,* $\mathsf{Sym}(R)$*,* $\mathsf{Tra}(R)$*, and* $\mathsf{Dis}(R, S)$*, **role assertions**, where, for each interpretation $\mathcal{I}$ and all $x,\ y,\ z \in \Delta^{\mathcal{I}}$, we have:*

$$\begin{array}{lll} \mathcal{I} \models \mathsf{Sym}(R) & \textit{if} & \langle x, y\rangle \in R^{\mathcal{I}} \textit{ implies } \langle y, x\rangle \in R^{I}; \\ \mathcal{I} \models \mathsf{Tra}(R) & \textit{if} & \langle x, y\rangle \in R^{\mathcal{I}} \textit{ and } \langle y, z\rangle \in R^{\mathcal{I}} \textit{ imply } \langle x, z\rangle \in R^{I}; \\ \mathcal{I} \models \mathsf{Ref}(R) & \textit{if} & Diag^{\mathcal{I}} \subseteq R^{\mathcal{I}}; \\ \mathcal{I} \models \mathsf{Irr}(R) & \textit{if} & R^{\mathcal{I}} \cap Diag^{\mathcal{I}} = \emptyset; \\ \mathcal{I} \models \mathsf{Dis}(R, S) & \textit{if} & R^{\mathcal{I}} \cap S^{\mathcal{I}} = \emptyset. \end{array}$$

Adding symmetric and transitive role assertions is a trivial move since both of these expressive means can be replaced by complex role inclusion axioms as follows: for the role assertion $\mathsf{Sym}(R)$ we can add to the Rbox, equivalently, the role inclusion axiom $R^- \dot{\sqsubseteq} R$, and, for the role assertion $\mathsf{Tra}(R)$, we can add to the Rbox, equivalently, $RR \dot{\sqsubseteq} R$. The proof of this should be obvious.

Thus, as far as expressivity is concerned, we can assume for convenience that no role assertions of the form $\mathsf{Tra}(R)$ or $\mathsf{Sym}(R)$ appear in $\mathcal{R}_a$, but that transitive and/or symmetric roles will be handled by the RIAs alone. In particular, notice that the addition of these role assertions can not trigger the Rbox to become non-regular.

The situation is different, however, for the other Rbox assertions. Neither reflexivity nor irreflexivity nor disjointness of roles can be enforced by role inclusion axioms. However, as we shall see later, reflexivity and irreflexivity of roles are closely related to the new concept $\exists R.\mathsf{Self}$.

In $\mathcal{SHIQ}$ (and $\mathcal{SHOIQ}$), the application of qualified number restrictions has to be restricted to certain roles, called *simple roles*, in order to preserve decidability (14). In the context of $\mathcal{SROIQ}$, the definition of *simple role* has to be slightly modified, and simple roles figure not only in qualified number restrictions, but in several other constructs as well. Intuitively, non-simple roles are those that are implied by the composition of roles.

Given a role hierarchy $\mathcal{R}_h$ and a set of role assertions $\mathcal{R}_a$ (without transitivity or symmetry assertions), the set of roles that are **simple in** $\mathcal{R} = \mathcal{R}_h \cup \mathcal{R}_a$ is inductively defined as follows:

- a role name is simple if it does not occur on the right hand side of a RIA in $\mathcal{R}_h$,
- an inverse role R^- is simple if R is, and

- if R occurs on the right hand side of a RIA in $\mathcal{R}_h$, then R is simple if, for each $w \dot{\sqsubseteq} R \in \mathcal{R}_h$, $w = S$ for a simple role S.

A set of role assertions $\mathcal{R}_a$ is called **simple** if all roles R, S appearing in role assertions of the form $\mathsf{Irr}(R)$ or $\mathsf{Dis}(R, S)$ are simple in $\mathcal{R}$. If $\mathcal{R}$ is clear from the context, we often use "simple" instead of "simple in $\mathcal{R}$".

Definition 4 (Role Box) *A $\mathcal{SROIQ}$-**role box** (Rbox for short) is a set $\mathcal{R} = \mathcal{R}_h \cup \mathcal{R}_a$, where $\mathcal{R}_h$ is a regular role hierarchy and $\mathcal{R}_a$ is a finite, simple set of role assertions.*

*An interpretation **satisfies a role box** $\mathcal{R}$ (written $\mathcal{I} \models \mathcal{R}$) if $\mathcal{I} \models R_h$ and $\mathcal{I} \models \phi$ for all role assertions $\phi \in R_a$. Such an interpretation is called a model of $\mathcal{R}$.*

2.2 Concepts and Inference Problems for $\mathcal{SROIQ}$

We are now ready to define the syntax and semantics of $\mathcal{SROIQ}$-concepts.

Definition 5 ($\mathcal{SROIQ}$ Concepts, Tboxes, and Aboxes)
*The set of $\mathcal{SROIQ}$-**concepts** is the smallest set such that*

- *every concept name (including nominals) and $\top, \bot$ are concepts, and,*
- *if C, D are concepts, R is a role (possibly inverse), S is a simple role (possibly inverse), and n is a non-negative integer, then $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall R.C$, $\exists R.C$, $\exists S.\mathsf{Self}$, $(\geqslant nS.C)$, and $(\leqslant nS.C)$ are also concepts.*

*A **general concept inclusion axiom** (GCI) is an expression of the form $C \dot{\sqsubseteq} D$ for two $\mathcal{SROIQ}$-concepts C and D. A **Tbox** $\mathcal{T}$ is a finite set of GCIs.*

*An **individual assertion** is of one of the following forms: $a : C$, $(a, b) : R$, $(a, b) : \neg S$, or $a \neq b$, for $a, b \in \mathbf{I}$ (the set of individual names), a (possibly inverse) role R, a (possibly inverse) simple role S, and a $\mathcal{SROIQ}$-concept C. A $\mathcal{SROIQ}$-**Abox** $\mathcal{A}$ is a finite set of individual assertions.*

Note that number restrictions $(\geqslant nS.C)$ and $(\leqslant nS.C)$, as well as the concept $\exists S.\mathsf{Self}$ and the disjointness and irreflexivity assertions for roles, $\mathsf{Dis}(R, S)$ and $\mathsf{Irr}(R)$, are all restricted to *simple* roles. In the case of number restrictions we mentioned the reason for this restriction already: without it, the satisfiability problem of $\mathcal{SHIQ}$-concepts is undecidable (14), even for a logic without inverse roles and with only *unqualifying* number restrictions (these are number restrictions of the form $(\geqslant nR.\top)$ and $(\leqslant nR.\top)$).

For $\mathcal{SROIQ}$ and the remaining restrictions to simple roles in concept expressions as well as role assertions, it is part of future work to determine which of these restrictions to simple roles is strictly necessary in order to preserve decidability or practicability. This restriction, however, allows a rather smooth integration of the new constructs into existing algorithms.

Definition 6 (Semantics and Inference Problems) *Given an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, concepts C, D, roles R, S, and non-negative integers n, the* ***extension of complex concepts*** *is defined inductively by the following equations, where $\sharp M$ denotes the cardinality of a set M, and concept names, roles, and nominals are interpreted as in Definition 1:*

$$
\begin{array}{lll}
\top^{\mathcal{I}} = \Delta^{\mathcal{I}}, \quad \bot^{\mathcal{I}} = \emptyset, \quad (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} & & \textit{(Booleans)} \\
(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}, \quad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} & & \textit{(Booleans)} \\
(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y.\langle x,y\rangle \in R^{\mathcal{I}} \textit{ and } y \in C^{\mathcal{I}}\} & & \textit{(exists restriction)} \\
(\exists R.\mathsf{Self})^{\mathcal{I}} = \{x \mid \langle x,x\rangle \in R^{\mathcal{I}}\} & & \textit{(}\exists R.\mathsf{Self}\textit{-concepts)} \\
(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y.\langle x,y\rangle \in R^{\mathcal{I}} \textit{ implies } y \in C^{\mathcal{I}}\} & & \textit{(value restriction)} \\
(\geqslant nR.C)^{\mathcal{I}} = \{x \mid \sharp\{y.\langle x,y\rangle \in R^{\mathcal{I}} \textit{ and } y \in C^{\mathcal{I}}\} \geqslant n\} & & \textit{(atleast restriction)} \\
(\leqslant nR.C)^{\mathcal{I}} = \{x \mid \sharp\{y.\langle x,y\rangle \in R^{\mathcal{I}} \textit{ and } y \in C^{\mathcal{I}}\} \leqslant n\} & & \textit{(atmost restriction)}
\end{array}
$$

An interpretation $\mathcal{I}$ is a ***model of a Tbox*** *$\mathcal{T}$ (written $\mathcal{I} \models \mathcal{T}$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for each GCI $C \mathrel{\dot\sqsubseteq} D$ in $\mathcal{T}$.*

A concept C is called ***satisfiable*** *if there is an interpretation $\mathcal{I}$ with $C^{\mathcal{I}} \neq \emptyset$. A concept D* ***subsumes*** *a concept C (written $C \sqsubseteq D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for each interpretation. Two concepts are* ***equivalent*** *(written $C \equiv D$) if they are mutually subsuming. The above inference problems can be defined w.r.t. a general role box $\mathcal{R}$ and/or a Tbox $\mathcal{T}$ in the usual way, i.e., by replacing* interpretation *with* model of $\mathcal{R}$ *and/or* $\mathcal{T}$.

For an interpretation $\mathcal{I}$, an element $x \in \Delta^{\mathcal{I}}$ is called an ***instance*** *of a concept C if $x \in C^{\mathcal{I}}$.*

An interpretation $\mathcal{I}$ ***satisfies*** *(is a model of)* ***an Abox*** *$\mathcal{A}$ ($\mathcal{I} \models \mathcal{A}$) if for all individual assertions $\phi \in \mathcal{A}$ we have $\mathcal{I} \models \phi$, where*

$$
\begin{array}{lll}
\mathcal{I} \models a{:}C & \textit{if} & a^{\mathcal{I}} \in C^{\mathcal{I}}; \\
\mathcal{I} \models a \neq b & \textit{if} & a^{\mathcal{I}} \neq b^{\mathcal{I}}; \\
\mathcal{I} \models (a,b){:}R & \textit{if} & \langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in R^{\mathcal{I}}; \\
\mathcal{I} \models (a,b){:}\neg R & \textit{if} & \langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \notin R^{\mathcal{I}}.
\end{array}
$$

An Abox $\mathcal{A}$ is ***consistent*** *with respect to an Rbox $\mathcal{R}$ and a Tbox $\mathcal{T}$ if there is a model $\mathcal{I}$ for $\mathcal{R}$ and $\mathcal{T}$ such that $\mathcal{I} \models \mathcal{A}$.*

2.3 Reduction of Inference Problems

For DLs that are closed under negation, subsumption and (un)satisfiability of concepts can be mutually reduced: $C \sqsubseteq D$ iff $C \sqcap \neg D$ is unsatisfiable, and C is unsatisfiable iff $C \sqsubseteq \bot$. Furthermore, a concept C is satisfiable iff the Abox $\{a : C\}$ (a a 'new' individual name) is consistent.

It is straightforward to extend these reductions to Rboxes and Tboxes. In contrast, the reduction of inference problems w.r.t. a Tbox to pure concept inference problems (possibly w.r.t. a role hierarchy), deserves special care: in (2; 20; 3), the *internalisation* of GCIs is introduced, a technique that realises exactly this reduction. For $\mathcal{SROIQ}$, this technique only needs to be slightly modified. We will show in a series of steps that $\mathcal{SROIQ}$ concept satisfiability of a concept C with respect to a triple $\langle \mathcal{A}, \mathcal{R}, \mathcal{T} \rangle$ of, respectively, a $\mathcal{SROIQ}$ Abox, Rbox, and Tbox, can be reduced to concept satisfiability of a concept C' with respect to an Rbox $\mathcal{R}'$, where the Rbox $\mathcal{R}'$ only contains role assertions of the form $\mathsf{Dis}(R, S)$, and the universal role U does not appear in C'.

While nominals can be used to 'internalise' the Abox, in order to eliminate the universal role, we use a 'simulated' universal role U', i.e., a reflexive, symmetric, and transitive super-role of all roles and their inverses appearing in $\langle \mathcal{A}, \mathcal{R}, \mathcal{T} \rangle$, and which, additionally, connects all nominals appearing in the input.

Thus, let C and $\langle \mathcal{A}, \mathcal{R}, \mathcal{T} \rangle$ be a $\mathcal{SROIQ}$ concept and Abox, Rbox, and Tbox, respectively. In a first step, we replace the Abox $\mathcal{A}$ with an Abox $\mathcal{A}'$ such that $\mathcal{A}'$ only contains individual assertions of the form $a : C$. In this regard, associate with every individual $a \in \mathbf{I}$ appearing in the input Abox $\mathcal{A}$ a new nominal o_a not appearing in $\mathcal{T}$ or C. Next, define $\mathcal{A}'$ by replacing every individual assertion in $\mathcal{A}$ of the form $(a, b) : R$ with $a : \exists R.o_b$, every $(a, b) : \neg R$ with $a : \forall R.\neg o_b$, and every $a \neq b$ with $a : \neg o_b$. Now, given C and $\mathcal{A}'$, define C' as follows:

$$C' := C \sqcap \bigsqcap_{a:C \in \mathcal{A}'} \exists U.(o_a \sqcap C),$$

where U is the universal role.

It should be clear that C is satisfiable with respect to $\langle \mathcal{A}, \mathcal{R}, \mathcal{T} \rangle$ if and only if C' is satisfiable with respect to $\langle \mathcal{R}, \mathcal{T} \rangle$. Thus we have:

Lemma 7 (Abox Elimination) *$\mathcal{SROIQ}$ concept satisfiability with respect to Aboxes, Rboxes, and Tboxes is polynomially reducible to $\mathcal{SROIQ}$ concept satisfiability with respect to Rboxes and Tboxes only.*

Hence, in the following we will assume that Aboxes have been eliminated. Next, although we have the 'real' universal role U present in the language, the following lemma shows how general concept inclusion axioms can be *internalised* while at the same time eliminating occurrences of the universal role U, using a simulated "universal" role U', that is, a transitive super-role of all roles (except U) occurring in $\mathcal{T}$ or $\mathcal{R}$ and their respective inverses. Furthermore, note that the universal role U is not allowed to appear in Rboxes.

Lemma 8 (Tbox and Universal Role Elimination) *Let C and D be concepts, $\mathcal{T}$ a Tbox, and $\mathcal{R} = \mathcal{R}_h \cup \mathcal{R}_a$ an Rbox. Let $U' \neq U$ be a role that does not occur in C, D, $\mathcal{T}$, or $\mathcal{R}$, and let C', D', and $\mathcal{T}'$ result from C, D, and $\mathcal{T}$ respectively, by replacing every occurrence of U by U'. We define*

$$C_{\mathcal{T}'} := \forall U'.\big(\bigsqcap_{C'_i \dot{\sqsubseteq} D'_i \in \mathcal{T}'} \neg C'_i \sqcup D'_i\big) \sqcap \big(\bigsqcap_{\mathbf{N} \ni o \in \mathcal{T} \cup C \cup D} \exists U'.o\big),$$

and set

$$\mathcal{R}_h^{U'} := \mathcal{R}_h \cup \{R \dot{\sqsubseteq} U' \mid R \textit{ occurs in } C',\ D',\ \mathcal{T}',\ \textit{or } \mathcal{R}\};$$

$$\mathcal{R}_a^{U'} := \mathcal{R}_a \cup \{\mathsf{Tra}(U'), \mathsf{Sym}(U'), \mathsf{Ref}(U')\}, \ \textit{and } \mathcal{R}_{U'} := \mathcal{R}_h^{U'} \cup \mathcal{R}_a^{U'}.$$

Then

- *C is satisfiable w.r.t. $\mathcal{T}$ and $\mathcal{R}$ iff $C' \sqcap C_{\mathcal{T}'}$ is satisfiable w.r.t. $\mathcal{R}_{U'}$.*
- *D subsumes C with respect to $\mathcal{T}$ and $\mathcal{R}$ iff $C' \sqcap \neg D' \sqcap C_{\mathcal{T}'}$ is unsatisfiable w.r.t. $\mathcal{R}_{U'}$.*

The proof of Lemma 8 is similar to the ones that can be found in (20; 2). Most importantly, it must be shown that (a): if a $\mathcal{SROIQ}$-concept C is satisfiable with respect to a Tbox $\mathcal{T}$ and an Rbox $\mathcal{R}$, then $C, \mathcal{T}, \mathcal{R}$ have a *weakly connected* model, i.e., a model which is a union of connected components, where each such component contains a nominal, and where any two elements of a connected component are connected by a role path over those roles occurring in C, $\mathcal{T}$ or $\mathcal{R}$, and (b): if y is reachable from x via a role path (possibly involving inverse roles), then $\langle x, y\rangle \in U'^{\mathcal{I}}$. These are easy consequences of the semantics and the definition of U' and $C_{\mathcal{T}'}$, which guarantees that all nominals are connected by U' links.

Now, note also that, instead of having a role assertion $\mathsf{Irr}(R) \in \mathcal{R}_a$, we can add, equivalently, the GCI $\top \dot{\sqsubseteq} \neg\exists R.\mathsf{Self}$ to $\mathcal{T}$, which can in turn be internalised. Likewise, instead of asserting $\mathsf{Ref}(R)$, we can, equivalently, add the GCI $\top \dot{\sqsubseteq} \exists R.\mathsf{Self}$ to $\mathcal{T}$. However, in the case of $\mathsf{Ref}(R)$ this replacement is only admissible for *simple* roles R and thus not possible (syntactically) in general.

Thus, using these equivalences (including the replacement of Rbox assertions of the form $\mathsf{Sym}(R)$ and $\mathsf{Tra}(R)$) and Lemmas 7 and 8, we arrive at the following theorem:

Theorem 9 (Reduction)

1. *Satisfiability and subsumption of $\mathcal{SROIQ}$-concepts w.r.t. Tboxes, Aboxes, and Rboxes, are polynomially reducible to (un)satisfiability of $\mathcal{SROIQ}$-concepts w.r.t. Rboxes.*

2. *W.l.o.g., we can assume that Rboxes do not contain role assertions of the form $\mathsf{Irr}(R)$, $\mathsf{Tra}(R)$, or $\mathsf{Sym}(R)$, and that the universal role is not used.*

With Theorem 9, all standard inference problems for $\mathcal{SROIQ}$-concepts and Aboxes can be reduced to the problem of determining the consistency of a $\mathcal{SROIQ}$-concept w.r.t. to an Rbox (both not containing the universal role), where we can assume w.l.o.g. that all role assertions in the Rbox are of the form $\mathsf{Ref}(R)$ or $\mathsf{Dis}(R, S)$—we call such an Rbox **reduced**.

3 $\mathcal{SROIQ}$ is Decidable

In this section, we show that $\mathcal{SROIQ}$ is decidable. We present a tableau-based algorithm that decides the consistency of a $\mathcal{SROIQ}$ concept w.r.t. a reduced Rbox, and therefore also all standard inference problems as discussed above, see Theorem 9. Therefore, in the following, by Rbox we always mean *reduced* Rbox.

The algorithm tries to construct, given a $\mathcal{SROIQ}$-concept C and an Rbox $\mathcal{R}$, a *tableau* for C and $\mathcal{R}$, that is, an abstraction of a model of C and $\mathcal{R}$. Given the appropriate notion of a tableau, it is then quite straightforward to prove that the algorithm is a decision procedure for $\mathcal{SROIQ}$-concept satisfiability with respect to Rboxes.

Before specifying this algorithm, we translate a role hierarchy $\mathcal{R}_h$ into non-deterministic automata which are used both in the definition of a tableau and in the tableau algorithm. Intuitively, an automaton is used to memorise the path between an object x that has to satisfy a concept of the form $\forall R.C$ and other objects, and then to determine which of these objects must satisfy C.[3]

For the following considerations, it is worthwhile to recall that, for a string $w = R_1 \dots R_m$ and R_i roles, $\mathsf{Inv}(w) = \mathsf{Inv}(R_m) \dots \mathsf{Inv}(R_1)$. The following Lemma is a direct consequence of the definition of the semantics.

Lemma 10 *If $\mathcal{I}$ is a model of $\mathcal{R}_h$ with $S^- \dot{\sqsubseteq} S \in \mathcal{R}_h$ and $w \dot{\sqsubseteq} S \in \mathcal{R}_h$, then $\mathsf{Inv}(w)^\mathcal{I} \subseteq S^\mathcal{I}$.*

3.1 Translating RIAs into Automata

The technique used in this chapter is identical to the one presented in (12), and repeated here only to make this paper self-contained. First, we will define, for a regular role hierarchy $\mathcal{R}_h$ and a (possibly inverse) role S occurring in $\mathcal{R}_h$, a non-deterministic finite automaton (NFA) $\mathcal{B}_S$ which captures all implications between (paths of) roles and S that are consequences of $\mathcal{R}_h$. To make this clear, before we define $\mathcal{B}_S$, we formulate the lemma which we are going to prove for it.

Proposition 11 *$\mathcal{I}$ is a model of $\mathcal{R}_h$ if and only if, for each (possibly inverse) role S occurring in $\mathcal{R}_h$, each word $w \in L(\mathcal{B}_S)$, and each $\langle x, y\rangle \in w^\mathcal{I}$, we have $\langle x, y\rangle \in S^\mathcal{I}$.*

In the following, we use NFAs with ε-transitions in a rather informal way (see, e.g., (7) for more details), e.g., we use $p \stackrel{R}{\rightarrow} q$ to denote that there is a transition from a state p to a state q with the letter R instead of introducing transition relations formally. The automata $\mathcal{B}_S$ are defined in three steps.

[3] This technique together with the relationship between automata and regular languages is the reason why we called these role hierarchies "regular".

Definition 12 *Let C be a $\mathcal{SROIQ}$-concept and $\mathcal{R}$ a reduced Rbox which is $\prec$-regular. For each role name R occurring in $\mathcal{R}$ or C, we first define the NFA $\mathcal{A}_R$ as follows: $\mathcal{A}_R$ contains a state i_R and a state f_R with the transition $i_R \stackrel{R}{\to} f_R$. The state i_R is the only initial state and f_R is the only final state. Moreover, for each $w \dot{\sqsubseteq} R \in \mathcal{R}$, $\mathcal{A}_R$ contains the following states and transitions:*

1. *if $w = RR$, then $\mathcal{A}_R$ contains $f_R \stackrel{\varepsilon}{\to} i_R$, and*
2. *if $w = R_1 \cdots R_n$ and $R_1 \neq R \neq R_n$, then $\mathcal{A}_R$ contains*
$$i_R \stackrel{\varepsilon}{\to} i_w \stackrel{R_1}{\to} f_w^1 \stackrel{R_2}{\to} f_w^2 \stackrel{R_3}{\to} \ldots \stackrel{R_n}{\to} f_w^n \stackrel{\varepsilon}{\to} f_R,$$
3. *if $w = RR_2 \cdots R_n$, then $\mathcal{A}_R$ contains*
$$f_R \stackrel{\varepsilon}{\to} i_w \stackrel{R_2}{\to} f_w^2 \stackrel{R_3}{\to} f_w^3 \stackrel{R_4}{\to} \ldots \stackrel{R_n}{\to} f_w^n \stackrel{\varepsilon}{\to} f_R,$$
4. *if $w = R_1 \cdots R_{n-1}R$, then $\mathcal{A}_R$ contains*
$$i_R \stackrel{\varepsilon}{\to} i_w \stackrel{R_1}{\to} f_w^1 \stackrel{R_2}{\to} f_w^2 \stackrel{R_3}{\to} \ldots \stackrel{R_{n-1}}{\to} f_w^{n-1} \stackrel{\varepsilon}{\to} i_R,$$

where all f_w^i, i_w are assumed to be distinct.

In the next step, we use a mirrored copy *of NFAs: this is a copy of an NFA in which we have carried out the following modifications: we*

- *make final states to non-final but initial states,*
- *make initial states to non-initial but final states,*
- *replace each transition $p \stackrel{S}{\to} q$ for S a (possibly inverse) role S with $q \stackrel{\mathsf{Inv}(S)}{\to} p$, and*
- *replace each transition $p \stackrel{\varepsilon}{\to} q$ with $q \stackrel{\varepsilon}{\to} p$.*

Secondly, we define the NFAs $\hat{\mathcal{A}}_R$ as follows:

- *if $R^- \dot{\sqsubseteq} R \notin \mathcal{R}$, then $\hat{\mathcal{A}}_R := \mathcal{A}_R$,*

- *if* $R^- \dot{\sqsubseteq} R \in \mathcal{R}$, *then* $\hat{\mathcal{A}}_R$ *is obtained as follows: first, take the disjoint union*[4] *of* $\mathcal{A}_S$ *with a mirrored copy of* $\mathcal{A}_S$. *Secondly, make* i_R *the only initial state,* f_R *the only final state. Finally, for* f'_R *the copy of* f_R *and* i'_R *the copy of* i_R, *add transitions* $i_R \stackrel{\varepsilon}{\rightarrow} f'_R$, $f'_R \stackrel{\varepsilon}{\rightarrow} i_R$, $i'_R \stackrel{\varepsilon}{\rightarrow} f_R$, *and* $f_R \stackrel{\varepsilon}{\rightarrow} i'_R$.

Thirdly, the NFAs $\mathcal{B}_R$ *are defined inductively over* $\prec$:

- *if* R *is minimal w.r.t.* $\prec$ *(i.e., there is no* R' *with* $R' \prec R$*), we set* $\mathcal{B}_R := \hat{\mathcal{A}}_R$.
- *otherwise,* $\mathcal{B}_R$ *is the disjoint union of* $\hat{A}_R$ *with a copy* $\mathcal{B}'_S$ *of* $\mathcal{B}_S$ *for each transition* $p \stackrel{S}{\rightarrow} q$ *in* $\hat{A}_R$ *with* $S \neq R$. *Moreover, for each such transition, we add* ε*-transitions from* p *to the initial state in* $\mathcal{B}'_S$ *and from the final state in* $\mathcal{B}'_S$ *to* q, *and we make* i_R *the only initial state and* f_R *the only final state in* $\mathcal{B}_R$.

Finally, the automaton $\mathcal{B}_{R^-}$ *is a mirrored copy of* $\mathcal{B}_R$.

Please note that the inductive definition of $\mathcal{B}_R$ is well-defined since the acyclic relation $\prec$ is used to restrict the dependencies between roles.

We have kept the construction of $\mathcal{B}_S$ as simple as possible. If one wants to construct an equivalent NFA without ε-transitions or which is deterministic, then there are well-known techniques to do this (7). Recall that elimination of ε-transitions can be carried out without increasing the number of an automaton's states, whereas determinisation might yield an exponential blow-up. However, as we will see later, this determinisation will happen anyway "on-the-fly" in the tableau algorithm, and thus has no influence on the complexity, see (12) for a discussion.

Lemma 13 *For* R *a role, the size of* $\mathcal{B}_R$ *is bounded exponentially in the depth*

$$d_{\mathcal{R}} := \max\{n \mid \text{there are } S_1 \prec \ldots \prec S_n, u_i, v_i \text{ with } u_i S_{i-1} v_i \dot{\sqsubseteq} S_i \in \mathcal{R}\}$$

and thus in the size of $\mathcal{R}$. *Moreover, there are* $\mathcal{R}$ *and* R *such that the number of states in* $\mathcal{B}_R$ *is* $2^{d_{\mathcal{R}}}$.

[4] A disjoint union of two automata is the disjoint union of their states, transition relations, etc.

In (12), certain further syntactic restrictions of role hierarchies were considered (there called *simple* role hierarchies) that avoid this exponential blow-up. We conjecture that without some such further restriction, this blow-up is unavoidable. Next, we will repeat a technical Lemma from (12) which we will use later, and refer the reader to (12) for its proof and the proof of Proposition 11.

Lemma 14 *1. $S \in L(\mathcal{B}_S)$ and, if $w \,\dot{\sqsubseteq}\, S \in \mathcal{R}$, then $w \in L(\mathcal{B}_S)$.*

2. If S is a simple role, then $L(\mathcal{B}_S) = \{R \mid R \,\underline{\sqsubseteq}^*\, S\}$.

3. If $\overleftarrow{\mathcal{A}}$ is a mirrored copy of an NFA $\mathcal{A}$, then $L(\overleftarrow{\mathcal{A}}) = \{\mathsf{Inv}(w) \mid w \in L(\mathcal{A})\}$.

3.2 A Tableau for $\mathcal{SROIQ}$

In the following, if not stated otherwise, C, D (possibly with subscripts) denote $\mathcal{SROIQ}$-concepts (not using the universal role), R, S (possibly with subscripts) roles, $\mathcal{R} = \mathcal{R}_h \cup \mathcal{R}_a$ an Rbox, and $\mathbf{R}_C$ the set of roles occurring in C and $\mathcal{R}$ together with their inverses. Furthermore, as noted in Theorem 12, we can (and will from now on) assume w.l.o.g. that all role assertions appearing in $\mathcal{R}_a$ are of the form $\mathsf{Dis}(R, S)$ or $\mathsf{Ref}(R)$.

We start by defining $\mathsf{fclos}(C_0, \mathcal{R})$, the *closure* of a concept C_0 w.r.t. a regular role hierarchy $\mathcal{R}$. Intuitively, this contains all relevant sub-concepts of C_0 together with universal value restrictions over sets of role paths described by an NFA. We use NFAs in universal value restrictions to memorise the path between an object that has to satisfy a value restriction and other objects. To do this, we "push" this NFA-value restriction along this path while the NFA gets "updated" with the path taken so far. For this "update", we use the following definition.

Definition 15 *For $\mathcal{B}$ an NFA and q a state of $\mathcal{B}$, $\mathcal{B}(q)$ denotes the NFA obtained from $\mathcal{B}$ by making q the (only) initial state of $\mathcal{B}$, and we use $q \xrightarrow{S} q' \in \mathcal{B}$ to denote that $\mathcal{B}$ has a transition $q \xrightarrow{S} q'$.*

Without loss of generality, we assume all concepts to be in NNF, that is, negation occurs only in front of concept names or in front of $\exists R.\mathsf{Self}$. Any $\mathcal{SROIQ}$-concept can easily be transformed into an equivalent one in NNF by pushing negations inwards using a combination of DeMorgan's laws and the following equivalences:

$$
\begin{array}{rclrcl}
\neg(\exists R.C) & \equiv & (\forall R.\neg C) & \neg(\forall R.C) & \equiv & (\exists R.\neg C) \\
\neg(\leqslant nR.C) & \equiv & (\geqslant(n+1)R.C) & \neg(\geqslant(n+1)R.C) & \equiv & (\leqslant nR.C) \\
 & & & \neg(\geqslant 0R.C) & \equiv & \bot
\end{array}
$$

We use $\dot{\neg}C$ for the NNF of $\neg C$. Obviously, the length of $\dot{\neg}C$ is linear in the length of C.

For a concept C_0, $\mathsf{clos}(C_0)$ is the smallest set that contains C_0 and that is closed under sub-concepts and $\dot{\neg}$. The set $\mathsf{fclos}(C_0, \mathcal{R})$ is then defined as follows:

$$\mathsf{fclos}(C_0,\mathcal{R}) := \mathsf{clos}(C_0) \cup \{\forall \mathcal{B}_S(q).D \mid \forall S.D \in \mathsf{clos}(C_0) \text{ and } \mathcal{B}_S \text{ has a state } q\}.$$

It is not hard to show and well-known that the size of $\mathsf{clos}(C_0)$ is linear in the size of C_0. For the size of $\mathsf{fclos}(C_0, \mathcal{R})$, we have seen in Lemma 13 that, for a role S, the size of $\mathcal{B}_S$ can be exponential in the depth of $\mathcal{R}$. Since there are at most linearly many concepts $\forall S.D$, this yields a bound for the cardinality of $\mathsf{fclos}(C_0, \mathcal{R})$ that is exponential in the depth of $\mathcal{R}$ and linear in the size of C_0.

Definition 16 (Tableau) *$T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ is a* tableau *for C_0 w.r.t. $\mathcal{R}$ iff*

- *$\mathbf{S}$ is a non-empty set;*
- *$\mathcal{L} : \mathbf{S} \to 2^{\mathsf{fclos}(C_0,\mathcal{R})}$ maps each element in $\mathbf{S}$ to a set of concepts;*
- *$\mathcal{E} : \mathbf{R}_{C_0} \to 2^{\mathbf{S}\times\mathbf{S}}$ maps each role to a set of pairs of elements in $\mathbf{S}$;*
- *$C_0 \in \mathcal{L}(s)$ for some $s \in \mathbf{S}$.*

Furthermore, for all $s, t \in \mathbf{S}$, $C, C_1, C_2 \in \mathsf{fclos}(C_0, \mathcal{R})$, $R, S \in \mathbf{R}_{C_0}$, and

$$S^T(s, C) := \{t \in \mathbf{S} \mid \langle s, t\rangle \in \mathcal{E}(S') \text{ for some } S' \in L(\mathcal{B}_S) \text{ and } C \in \mathcal{L}(t)\},$$

the tableau T satisfies:

(P1a) *if* $C \in \mathcal{L}(s)$, *then* $\neg C \notin \mathcal{L}(s)$ (C *atomic or* $\exists R.\mathsf{Self}$),
(P1b) $\top \in \mathcal{L}(s)$, *and* $\bot \notin \mathcal{L}(s)$, *for all* s,
(P1c) *if* $\exists R.\mathsf{Self} \in \mathcal{L}(s)$, *then* $\langle s, s\rangle \in \mathcal{E}(R)$,
(P1d) *if* $\neg\exists R.\mathsf{Self} \in \mathcal{L}(s)$, *then* $\langle s, s\rangle \notin \mathcal{E}(R)$,
(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* $C_2 \in \mathcal{L}(s)$,
(P4a) *if* $\forall\mathcal{B}(p).C \in \mathcal{L}(s)$, $\langle s, t\rangle \in \mathcal{E}(S)$, *and* $p \xrightarrow{S} q \in \mathcal{B}(p)$, *then* $\forall\mathcal{B}(q).C \in \mathcal{L}(t)$,
(P4b) *if* $\forall\mathcal{B}.C \in \mathcal{L}(s)$ *and* $\varepsilon \in L(\mathcal{B})$, *then* $C \in \mathcal{L}(s)$,
(P5) *if* $\exists S.C \in \mathcal{L}(s)$, *then there is some* t *with* $\langle s, t\rangle \in \mathcal{E}(S)$ *and* $C \in \mathcal{L}(t)$,
(P6) *if* $\forall S.C \in \mathcal{L}(s)$, *then* $\forall\mathcal{B}_S.C \in \mathcal{L}(s)$,
(P7) $\langle x, y\rangle \in \mathcal{E}(R)$ *iff* $\langle y, x\rangle \in \mathcal{E}(\mathsf{Inv}(R))$,
(P8) *if* $({\leqslant}nS.C) \in \mathcal{L}(s)$, *then* $\sharp S^T(s, C) \leqslant n$,
(P9) *if* $({\geqslant}nS.C) \in \mathcal{L}(s)$, *then* $\sharp S^T(s, C) \geqslant n$,
(P10) *if* $({\leqslant}nS.C) \in \mathcal{L}(s)$ *and* $\langle s, t\rangle \in \mathcal{E}(S)$, *then* $C \in \mathcal{L}(t)$ *or* $\dot{\neg} C \in \mathcal{L}(t)$,
(P11) *if* $\mathsf{Dis}(R, S) \in \mathcal{R}_a$, *then* $\mathcal{E}(R) \cap \mathcal{E}(S) = \emptyset$,
(P12) *if* $\mathsf{Ref}(R) \in \mathcal{R}_a$, *then* $\langle s, s\rangle \in \mathcal{E}(R)$ *for all* $s \in \mathbf{S}$,
(P13) *if* $\langle s, t\rangle \in \mathcal{E}(R)$ *and* $R \sqsubseteq\!\!\!{*}\, S$, *then* $\langle s, t\rangle \in \mathcal{E}(S)$,
(P14a) $o \in \mathcal{L}(s)$ *for some* $s \in \mathbf{S}$, *for each* $o \in \mathbf{N} \cap \mathsf{fclos}(C_0, \mathcal{R})$,
(P14b) *if* $o \in \mathcal{L}(s) \cap \mathcal{L}(t)$ *for some* $o \in \mathbf{N}$, *then* $s = t$.

Theorem 17 (Tableau) *A* $\mathcal{SROIQ}$*-concept* C_0 *is satisfiable w.r.t. a reduced Rbox* $\mathcal{R}$ *iff there exists a tableau for* C_0 *w.r.t.* $\mathcal{R}$.

Proof: For the *if* direction, let $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ be a tableau for C_0 w.r.t. $\mathcal{R}$. We extend the relational structure of T and then prove that this indeed gives a model.

More precisely, a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of C_0 and $\mathcal{R}$ can be defined as follows: we set $\Delta^{\mathcal{I}} := \mathbf{S}$, $C^{\mathcal{I}} := \{s \mid C \in \mathcal{L}(s)\}$ for concept names C in $\mathsf{fclos}(C_0, \mathcal{R})$, where (P14a) and (P14b) guarantee that nominals are indeed interpreted as singleton sets, and for roles names $R \in \mathbf{R}_{C_0}$, we set

$$R^{\mathcal{I}} := \{\langle s_0, s_n\rangle \in (\Delta^{\mathcal{I}})^2 \mid \text{exists } s_1, \ldots, s_{n-1} \text{ with } \langle s_i, s_{i+1}\rangle \in \mathcal{E}(S_{i+1}) \text{ for } 0 \leq i \leq n-1 \text{ and } S_1 \cdots S_n \in L(\mathcal{B}_R)\}$$

The semantics of complex concepts is given through the definition of the $\mathcal{SROIQ}$-semantics. Due to Lemma 14.3 and (P7), the semantics of inverse roles can either be given directly as for role names, or by setting $(R^-)^\mathcal{I} := \{\langle y,x\rangle \mid \langle x,y\rangle \in R^\mathcal{I}\}$.

We have to show that $\mathcal{I}$ is a model of $\mathcal{R}$ and C_0. We begin by showing that $\mathcal{I} \models \mathcal{R}$. First, we look at role assertions. Remember that we assumed that $\mathcal{R}$ is reduced, and thus we only have to deal with role disjointness assertions of the form $\mathsf{Dis}(R,S)$ and reflexivity assertions of the form $\mathsf{Ref}(R)$.

Consider an assertion $\mathsf{Dis}(R,S) \in \mathcal{R}$. By definition of $\mathcal{SROIQ}$-Rboxes, both R and S are simple roles, and (P11) implies $\mathcal{E}(R) \cap \mathcal{E}(S) = \emptyset$. Moreover, we have, by definition of $\mathcal{I}$, Lemma 14.2, (P7), and (P13) that, for T a simple role, $T^\mathcal{I} = \mathcal{E}(T)$. Hence $R^\mathcal{I} \cap S^\mathcal{I} = \emptyset$. Next, if $\mathsf{Ref}(R) \in \mathcal{R}_a$ for R a possibly non-simple role, we have immediately, by (P12) and $R \in L(\mathcal{B}_R)$, that $Diag^\mathcal{I} \subseteq R^\mathcal{I}$, and thus $\mathcal{I}$ satisfies each role assertion in $\mathcal{R}_a$.

Next, we show that $\mathcal{I} \models \mathcal{R}_h$. Due to Proposition 11, it suffices to prove that, for each (possibly inverse) role S, each word $w \in L(\mathcal{B}_S)$, and each $\langle x,y\rangle \in w^\mathcal{I}$, we have $\langle x,y\rangle \in S^\mathcal{I}$.

Let $w \in L(\mathcal{B}_S)$ and $\langle x,y\rangle \in w^\mathcal{I}$. For $w = S_1 \ldots S_n$, this implies the existence of y_i such that $y_0 = x$, $y_n = y$, and $\langle y_{i-1}, y_i\rangle \in S_i^\mathcal{I}$ for each $1 \le i \le n$. For each i, we define a word w_i as follows:

- if $\langle y_{i-1}, y_i\rangle \in \mathcal{E}(S_i)$, then set $w_i := S_i$.
- otherwise, there is some $v_i = T_1^{(i)} \ldots T_{n_i}^{(i)} \in L(\mathcal{B}_{S_i})$ and there are $y_j^{(i)}$ such that $y_{i-1} = y_0^{(i)}$, $y_i = y_{n_i}^{(i)}$, and $\langle y_{j-1}^{(i)}, y_j^{(i)}\rangle \in \mathcal{E}(T_j^{(i)})$ for each $1 \le j \le n_i$. In this case, we set $w_i := v_i$.

Let $\hat{w} := w_1 \ldots w_n$. By construction of $\mathcal{B}_S$ from $\hat{\mathcal{A}}_S$, $w \in L(\mathcal{B}_S)$ implies that $\hat{w} \in L(\mathcal{B}_S)$. For $\hat{w} = U_1 \ldots U_{n'}$, we can thus re-name the y_i and $y_j^{(i)}$ to z_i such that we have $z_0 = x$, $z_n = y$, and $\langle z_{i-1}, z_i\rangle \in \mathcal{E}(U_i)$. Hence, by definition of $\cdot^\mathcal{I}$, we have $\langle x,y\rangle \in S^\mathcal{I}$.

Secondly, we prove that $\mathcal{I}$ is a model of C_0. We show that $C \in \mathcal{L}(s)$ implies $s \in C^\mathcal{I}$ for each $s \in \mathbf{S}$ and each $C \in \mathsf{fclos}(\mathcal{A},\mathcal{R})$. This proof can be given by induction on the length of concepts, where we count neither negation nor integers in number restrictions. The only interesting cases are $C = ({\leqslant} nS.E)$, $C = \forall S.E$, and $C = (\neg)\exists R.\mathsf{Self}$ (for the other cases, see (17; 11)):

- If $({\leqslant} nS.E) \in \mathcal{L}(s)$, then ($\mathsf{P}$8) implies that $\# S^T(s,E) \le n$. Moreover, since S is simple, Lemma 14.2 implies that $L(\mathcal{B}_S) = \{S' \mid S' \mathrel{\underline{\circledast}} S\}$, and ($\mathsf{P}$13) implies that $S^\mathcal{I} = \mathcal{E}(S)$. Hence ($\mathsf{P}$10) implies that, for all t, if $\langle s,t\rangle \in S^{\mathcal{I}}$, then $E \in \mathcal{L}(t)$ or $\dot\neg E \in \mathcal{L}(t)$. By induction $E^{\mathcal{I}} = \{t \mid E \in \mathcal{L}(t)\}$, and thus $s \in (\leqslant nS.E)^{\mathcal{I}}$.

- Let $\forall S.E \in \mathcal{L}(s)$ and $\langle s,t\rangle \in S^{\mathcal{I}}$. From (P6) we have that $\forall \mathcal{B}_S.E \in \mathcal{L}(s)$. By definition of $S^{\mathcal{I}}$, there are $S_1 \dots S_n \in L(\mathcal{B}_S)$ and s_i with $s = s_0$, $t = s_n$, and $\langle s_{i-1}, s_i\rangle \in \mathcal{E}(S_i)$. Applying (P4a) n times, this yields $\forall \mathcal{B}_S(q).E \in \mathcal{L}(t)$ for q a final state of $\mathcal{B}_S$. Thus (P4b) implies that $E \in \mathcal{L}(t)$. By induction, $t \in E^{\mathcal{I}}$, and thus $s \in (\forall S.E)^{\mathcal{I}}$.
- Let $\exists R.\mathsf{Self} \in \mathcal{L}(s)$. Then, by (P1c), $\langle s,s\rangle \in \mathcal{E}(R)$ and, since $R \in L(B_R)$ and by definition of $\mathcal{I}$, we have $\langle s,s\rangle \in R^{\mathcal{I}}$. It follows that $s \in (\exists R.\mathsf{Self})^{\mathcal{I}}$.
- Let $\neg\exists R.\mathsf{Self} \in \mathcal{L}(s)$. Then, by (P1d), $\langle s,s\rangle \notin \mathcal{E}(R)$. Since R is a simple role by definition, we have, as shown above, $R^{\mathcal{I}} = \mathcal{E}(R)$. Hence $\langle s,s\rangle \notin R^{\mathcal{I}}$, and so $s \in (\neg\exists R.\mathsf{Self})^{\mathcal{I}}$.

For the converse, suppose $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of C_0 w.r.t. $\mathcal{R}$. We define a tableau $T = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ for C_0 and $\mathcal{R}$ as follows:

$$\begin{array}{rcl}
\mathbf{S} & := & \Delta^{\mathcal{I}}, \\
\mathcal{E}(R) & := & R^{\mathcal{I}}, \text{ and} \\
\mathcal{L}(s) & := & \{C \in \mathsf{fclos}(C_0,\mathcal{R}) \mid s \in C^{\mathcal{I}}\} \cup \\
 & & \{\forall \mathcal{B}_S.C \mid \forall S.C \in \mathsf{fclos}(C_0,\mathcal{R}) \text{ and } s \in (\forall S.C)^{\mathcal{I}}\} \cup \\
 & & \{\forall \mathcal{B}_R(q).C \in \mathsf{fclos}(C_0,\mathcal{R}) \mid \text{ for all } S_1 \cdots S_n \in L(\mathcal{B}_R(q)), \\
 & & \qquad s \in (\forall S_1.\forall S_2.\cdots\forall S_n.C)^{\mathcal{I}} \text{ and} \\
 & & \qquad \text{if } \varepsilon \in L(\mathcal{B}_R(q)), \text{ then } s \in C^{\mathcal{I}}\}
\end{array}$$

We have to show that T satisfies each (Pi). We restrict our attention to the only new cases.

For (P6), if $\forall S.C \in \mathcal{L}(s)$, then $s \in (\forall S.C)^{\mathcal{I}}$ and thus $\forall \mathcal{B}_S.C \in \mathcal{L}(s)$ by definition of T.

For (P4a), let $\forall \mathcal{B}(p).C \in \mathcal{L}(s)$ and $\langle s,t\rangle \in \mathcal{E}(S) = S^{\mathcal{I}}$. Assume that there is a transition $p \xrightarrow{S} q$ in $\mathcal{B}(p)$ and $\forall \mathcal{B}(q).C \notin \mathcal{L}(t)$. By definition of T, this can have two reasons:

- there is a word $S_2 \dots S_n \in L(\mathcal{B}(q))$ and $t \notin (\forall S_2.\dots\forall S_n.C)^{\mathcal{I}}$. However, this implies that $SS_2 \dots S_n \in L(\mathcal{B}(p))$ and thus that we have $s \in (\forall S.\forall S_2.\dots\forall S_n.C)^{\mathcal{I}}$, which contradicts, together with $\langle s,t\rangle \in S^{\mathcal{I}}$, the definition of the semantics of $\mathcal{SROIQ}$ concepts.

- $\varepsilon \in L(\mathcal{B}(q))$ and $t \notin C^{\mathcal{I}}$. This implies that $S \in L(\mathcal{B}(p))$ and thus contradicts $s \in (\forall S.C)^{\mathcal{I}}$.

Hence $\forall \mathcal{B}(q).C \notin \mathfrak{L}(t)$.

For (P4b), $\varepsilon \in L(\mathcal{B}(p))$ implies $s \in C^{\mathcal{I}}$ by definition of T, and thus $C \in \mathfrak{L}(s)$.

Finally, (P11)–(P14b) follow immediately from the definition of the semantics.

□

3.3 The Tableau Algorithm

In this section, we present a terminating, sound, and complete tableau algorithm that decides consistency of $\mathcal{SROIQ}$-concepts not using the universal role w.r.t. reduced Rboxes, and thus, using Theorem 9, also concept satisfiability w.r.t. Rboxes, Tboxes and Aboxes.

We first define the underlying data structures and corresponding operations. For more detailed comments about the intuitions underlying these definitions, consult (13).

The algorithm generates a *completion graph*, a structure that, if complete and clash-free, can be unravelled to a (possibly infinite) tableau for the input concept and Rbox. Moreover, it is shown that the algorithm returns a complete and clash-free completion graph for C_0 and $\mathcal{R}$ if and only if there exists a tableau for C_0 and $\mathcal{R}$, and thus with Lemma 17, if and only if the concept C_0 is satisfiable w.r.t. $\mathcal{R}$.

As usual, in the presence of transitive roles, *blocking* is employed to ensure termination of the algorithm. In the additional presence of inverse roles, blocking is *dynamic*, i.e., blocked nodes (and their sub-branches) can be un-blocked and blocked again later. In the further, additional presence of number restrictions, *pairs* of nodes are involved in the definition of blocking rather than single nodes (17). The blocking conditions as they are presented here are, clearly, too strict. As a consequence, blocking may occur later than necessary, and thus we end up with a search space that is larger than necessary. In (11), it was shown how to loosen the blocking condition for $\mathcal{SHIQ}$ while retaining correctness of the algorithm. Here, we focus on the decidability of $\mathcal{SROIQ}$, and defer a similar loosening for $\mathcal{SROIQ}$ to future work.

Definition 18 (Completion Graph) *Let $\mathcal{R}$ be a reduced Rbox, let C_0 be a $\mathcal{SROIQ}$-concept in NNF not using the universal role, and let $\mathbf{N}$ be the set of nominals. A **completion graph** for C_0 with respect to $\mathcal{R}$ is a directed graph $\mathbf{G} = (V, E, \mathfrak{L}, \neq)$ where each node $x \in V$ is labelled with a set*

$$\mathfrak{L}(x) \subseteq \mathsf{fclos}(C_0, \mathcal{R}) \cup \mathbf{N} \cup \{(\leqslant mR.C) \mid (\leqslant nR.C) \in \mathsf{fclos}(C_0, \mathcal{R}) \text{ and } m \leq n\}$$

and each edge $\langle x, y\rangle \in E$ is labelled with a set of role names $\mathfrak{L}(\langle x, y\rangle)$ containing (possibly inverse) roles occurring in C_0 or $\mathcal{R}$. Additionally, we keep track of inequalities between nodes of the graph with a symmetric binary relation $\neq$ between the nodes of $\mathbf{G}$.

In the following, we often use $R \in \mathfrak{L}(\langle x, y\rangle)$ as an abbreviation for $\langle x, y\rangle \in E$ and $R \in \mathfrak{L}(\langle x, y\rangle)$.

*If $\langle x, y\rangle \in E$, then y is called a **successor** of x and x is called a **predecessor** of y. **Ancestor** is the transitive closure of predecessor, and **descendant** is the transitive closure of successor. A node y is called an R-**successor** of a node x if, for some R' with $R' \,\underline{\sqsubseteq}^{*}\, R$, $R' \in \mathfrak{L}(\langle x, y\rangle)$. A node y is called a **neighbour** (R-**neighbour**) of a node x if y is a successor (R-successor) of x or if x is a successor ($\mathsf{Inv}(R)$-successor) of y.*

For a role S and a node x in $\mathbf{G}$, we define the set of x's S-neighbours with C in their label, $S^{\mathbf{G}}(x, C)$, as follows:

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

*$\mathbf{G}$ is said to **contain a clash** if there are nodes x and y such that*

1. *$\bot \in \mathfrak{L}(x)$, or*
2. *for some concept name A, $\{A, \neg A\} \subseteq \mathfrak{L}(x)$, or*
3. *x is an S-neighbour of x and $\neg\exists S.\mathsf{Self} \in \mathfrak{L}(x)$, or*
4. *there is some $\mathsf{Dis}(R, S) \in \mathcal{R}_a$ and y is an R- and an S-neighbour of x, or*
5. *there is some concept $(\leqslant nS.C) \in \mathfrak{L}(x)$ and $\{y_0, \ldots, y_n\} \subseteq S^{\mathbf{G}}(x, C)$ with $y_i \neq y_j$ for all $0 \leq i < j \leq n$, or*
6. *for some $o \in \mathbf{N}$, $x \neq y$ and $o \in \mathfrak{L}(x) \cap \mathfrak{L}(y)$.*

If $o_1, \ldots, o_\ell$ are all the nominals occurring in C_0, then the tableau algorithm starts with the completion graph $\mathbf{G} = (\{r_0, r_1 \ldots, r_\ell\}, \emptyset, \mathcal{L}, \emptyset)$ with $\mathcal{L}(r_0) = \{C_0\}$ and $\mathcal{L}(r_i) = \{o_i\}$ for $1 \leq i \leq \ell$. $\mathbf{G}$ is then expanded by repeatedly applying the expansion rules given in Figure 1, stopping if a clash occurs.

Before describing the tableau algorithm in more detail, we define some terms and operations used in the (application of the) expansion rules:

Nominal Nodes and Blockable Nodes We distinguish two types of nodes in $\mathbf{G}$, **nominal nodes** and **blockable nodes**. A node x is a nominal node if $\mathcal{L}(x)$ contains a nominal. A node that is not a nominal node is a blockable node. A nominal $o \in \mathbf{N}$ is said to be **new in G** if no node in $\mathbf{G}$ has o in its label.
Comment: like ABox individuals (16), nominal nodes can be arbitrarily interconnected. In contrast, blockable nodes are only found in tree-like structures rooted in nominal nodes (or in r_0); a branch of such a tree may simply end, possibly with a **blocked** node (defined below) as a leaf, or have an edge leading to a nominal node. In case a branch ends in a blocked node, we use standard *unravelling* to construct a tableau from the completion graph, and thus the resulting tableau will contain infinitely many copies of the nodes on the path from the blocking node to the blocked node. This is why there can be no nominal nodes on this path.

In the *NN*-rule, we use *new* nominals to create new nominal nodes—intuitively, to fix the identity of certain, constrained neighbours of nominal nodes. As we will show, it is possible to fix an upper bound on the number of nominal nodes that can be generated in a given completion graph; this is crucial for termination of the construction, given that blocking cannot be applied to nominal nodes.

Blocking A node x is **label blocked** if it has ancestors x', y and y' such that

1. x is a successor of x' and y is a successor of y',
2. y, x and all nodes on the path from y to x are blockable,
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 say that y **blocks** x. A node is **blocked** if either it is label blocked or it is blockable and its predecessor is blocked; if the predecessor of a safe node x is blocked, then we say that x is **indirectly blocked**.
Comment: blocking is defined exactly as for $\mathcal{SHIQ}$, with the only difference that, in the presence of nominals, we must take care that none of the nodes between a blocking and a blocked one is a nominal node.

Generating and Shrinking Rules and Safe Neighbours The $\geqslant$-, $\exists$- and NN-rules are called **generating rules**, and the $\leqslant$- and the o-rule are called **shrinking rules**. An R-neighbour y of a node x is **safe** if (i) x is blockable or if (ii) x is a nominal node and y is not blocked.
Comment: generating rules add new nodes to the completion graph, whereas shrinking rules remove nodes—they merge all information concerning one node into another one (e.g., to satisfy atmost number restrictions), and then remove the former node. We need the safety of R-neighbours to ensure that enough R-neighbours for nominal nodes are generated.

Pruning When a node y is **merged** into a node x, we "prune" the completion graph by removing y and, recursively, all blockable successors of y. More precisely, pruning a node y (written $\mathsf{Prune}(y)$) in $\mathbf{G} = (V, E, \mathcal{L}, \dot{\neq})$ yields a graph that is obtained from $\mathbf{G}$ as follows:

1. for all successors z of y, remove $\langle y, z\rangle$ from E and, if z is blockable, $\mathsf{Prune}(z)$;
2. remove y from V.

Merging In some rules, we "merge" one node into another node. Intuitively, when we merge a node y into a node x, we add $\mathcal{L}(y)$ to $\mathcal{L}(x)$, "move" all the edges leading *to* y so that they lead to x and "move" all the edges leading from y to nominal nodes so that they lead from x to the same nominal nodes; we then remove y (and blockable sub-trees below y) from the completion graph. More precisely, merging a node y into a node x (written $\mathsf{Merge}(y, x)$) in $\mathbf{G} = (V, E, \mathcal{L}, \dot{\neq})$ yields a graph that is obtained from $\mathbf{G}$ as follows:

1. for all nodes z such that $\langle z, y\rangle \in E$

(a) if $\{\langle x,z\rangle,\langle z,x\rangle\}\cap E=\emptyset$, then add $\langle z,x\rangle$ to E and set $\mathcal{L}(\langle z,x\rangle)=\mathcal{L}(\langle z,y\rangle)$,

(b) if $\langle z,x\rangle\in E$, then set $\mathcal{L}(\langle z,x\rangle)=\mathcal{L}(\langle z,x\rangle)\cup\mathcal{L}(\langle z,y\rangle)$,

(c) if $\langle x,z\rangle\in E$, then set $\mathcal{L}(\langle x,z\rangle)=\mathcal{L}(\langle x,z\rangle)\cup\{\mathsf{Inv}(S)\mid S\in\mathcal{L}(\langle z,y\rangle)\}$, and

(d) remove $\langle z,y\rangle$ from E;

2. for all nominal nodes z such that $\langle y,z\rangle\in E$

   (a) if $\{\langle x,z\rangle,\langle z,x\rangle\}\cap E=\emptyset$, then add $\langle x,z\rangle$ to E and set $\mathcal{L}(\langle x,z\rangle)=\mathcal{L}(\langle y,z\rangle)$,

   (b) if $\langle x,z\rangle\in E$, then set $\mathcal{L}(\langle x,z\rangle)=\mathcal{L}(\langle x,z\rangle)\cup\mathcal{L}(\langle y,z\rangle)$,

   (c) if $\langle z,x\rangle\in E$, then set $\mathcal{L}(\langle z,x\rangle)=\mathcal{L}(\langle z,x\rangle)\cup\{\mathsf{Inv}(S)\mid S\in\mathcal{L}(\langle y,z\rangle)\}$, and

   (d) remove $\langle y,z\rangle$ from E;

3. set $\mathcal{L}(x)=\mathcal{L}(x)\cup\mathcal{L}(y)$;

4. add $x\neq z$ for all z such that $y\neq z$; and

5. $\mathsf{Prune}(y)$.

If y was merged into x, we call x a **direct heir** of y, and we use being an **heir** of another node for the transitive closure of being a "direct heir".
Comment: merging is the generalisation of what is often done to satisfy an atmost number restriction for a node x in case that x has too many neighbours. However, since we might need to merge nominal nodes that are related in some arbitrary, non-tree-like way, merging gets slightly more tricky since we must take care of all incoming and outgoing edges. The usage of "heir" is quite intuitive since, after y has been merged into x, x has "inherited" all of y's properties, i.e., its label, its inequalities, and its incoming and outgoing edges (except for any outgoing edges removed by Prune).

Level (of Nominal Nodes) Let $o_1,\dots,o_\ell$ be all the nominals occurring in the input concept D. We define the *level* of a node inductively as follows:

- each (nominal) node x with an $o_i\in\mathcal{L}(x)$, $1\leq i\leq\ell$, is of level 0, and

- a nominal node x is of level i if x is not of some level $j < i$ and x has a neighbour that is of level $i - 1$.

Comment: if a node with a lower level is merged into another node, the level of the latter node may be reduced, but it can never be increased because Merge preserves all edges connecting nominal nodes. The completion graph initially contains only level 0 nodes.

Strategy (of Rule Application) the expansion rules are applied according to the following strategy:

1. the o-rule is applied with highest priority,
2. next, the $\leqslant$- and the NN-rule are applied, and they are applied first to nominal nodes with lower levels (before they are applied to nodes with higher levels). In case they are both applicable to the same node, the NN-rule is applied first.
3. all other rules are applied with a lower priority.

Comment: this strategy is necessary for termination, and in particular to fix an upper bound on the number of applications of the NN-rule. The general idea is to apply shrinking rules before any other rules (with the exception that the NN-rule is applied to a node *before* applying the $\leqslant$-rule to it), and to apply these "crucial" rules to lower level nodes before applying them to higher level nodes.

We are now ready to finish the description of the tableau algorithm:

A completion graph is **complete** if it contains a clash, or when none of the rules is applicable. If the expansion rules can be applied to C_0 and $\mathcal{R}$ in such a way that they yield a complete, clash-free completion graph, then the algorithm returns "C_0 is *satisfiable* w.r.t. $\mathcal{R}$", and "C_0 is *unsatisfiable* w.r.t. $\mathcal{R}$" otherwise.

3.4 Termination, Soundness, and Completeness

All but the Self–Ref-rule have been used before for fragments of $\mathcal{SROIQ}$, see (14; 11; 12), and the three $\forall_i$-rules are the obvious counterparts to the tableau conditions (P4a), (P4b), and (P6) of (12).

$\sqcap$-rule:	if $C_1 \sqcap C_2 \in \mathcal{L}(x)$, x is not indirectly blocked, and $\{C_1, C_2\} \not\subseteq \mathcal{L}(x)$, then $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{C_1, C_2\}$
$\sqcup$-rule:	if $C_1 \sqcup C_2 \in \mathcal{L}(x)$, x is not indirectly blocked, and $\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$ then $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{E\}$ for some $E \in \{C_1, C_2\}$
$\exists$-rule:	if $\exists S.C \in \mathcal{L}(x)$, x is not blocked, and 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 $\mathcal{L}(y) := \{C\}$
Self–Ref-rule:	if $\exists S.\mathsf{Self} \in \mathcal{L}(x)$ or $\mathsf{Ref}(S) \in \mathcal{R}_a$, x is not blocked, and $S \notin \mathcal{L}(\langle x, x\rangle)$ then add an edge $\langle x, x\rangle$ if it does not yet exist, and set $\mathcal{L}(\langle x, x\rangle) \longrightarrow \mathcal{L}(\langle x, x\rangle) \cup \{S\}$
$\forall_1$-rule:	if $\forall S.C \in \mathcal{L}(x)$, x is not indirectly blocked, and $\forall \mathcal{B}_S.C \notin \mathcal{L}(x)$ then $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{\forall \mathcal{B}_S.C\}$
$\forall_2$-rule:	if $\forall \mathcal{B}(p).C \in \mathcal{L}(x)$, x is not indirectly blocked, $p \xrightarrow{S} q$ in $\mathcal{B}(p)$, and there is an S-neighbour y of x with $\forall \mathcal{B}(q).C \notin \mathcal{L}(y)$, then $\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\forall \mathcal{B}(q).C\}$
$\forall_3$-rule:	if $\forall \mathcal{B}.C \in \mathcal{L}(x)$, x is not indirectly blocked, $\varepsilon \in L(\mathcal{B})$, and $C \notin \mathcal{L}(x)$ then $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{C\}$
choose-rule:	if $(\leqslant nS.C) \in \mathcal{L}(x)$, x is not indirectly blocked, and there is an S-neighbour y of x with $\{C, \dot\neg C\} \cap \mathcal{L}(y) = \emptyset$ then $\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{E\}$ for some $E \in \{C, \dot\neg C\}$
$\geqslant$-rule:	if 1. $(\geqslant nS.C) \in \mathcal{L}(x)$, x is not blocked, and 2. there are not n safe S-neighbours $y_1, \ldots, y_n$ of x with $C \in \mathcal{L}(y_i)$ and $y_i \not\doteq y_j$ 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 $y_i \not\doteq y_j$ for $1 \leq i < j \leq n$.
$\leqslant$-rule:	if 1. $(\leqslant nS.C) \in \mathcal{L}(z)$, z is not indirectly blocked, and 2. $\sharp S^{\mathbf{G}}(z, C) > n$ and there are two S-neighbours x, y of z with $C \in \mathcal{L}(x) \cap \mathcal{L}(y)$, and not $x \not\doteq y$ then 1. if x is a nominal node, then $\mathsf{Merge}(y, x)$ 2. else if y is a nominal node or an ancestor of x, then $\mathsf{Merge}(x, y)$ 3. else $\mathsf{Merge}(y, x)$
o-rule:	if for some $o \in N_I$ there are 2 nodes x, y with $o \in \mathcal{L}(x) \cap \mathcal{L}(y)$ and not $x \not\doteq y$ then $\mathsf{Merge}(x, y)$
NN-rule:	if 1. $(\leqslant nS.C) \in \mathcal{L}(x)$, x is a nominal node, and there is a blockable S-neighbour y of x such that $C \in \mathcal{L}(y)$ and x is a successor of y, 2. there is no m such that $1 \leqslant m \leqslant n$, $(\leqslant mS.C) \in \mathcal{L}(x)$, and there exist m nominal S-neighbours $z_1, \ldots, z_m$ of x with $C \in \mathcal{L}(z_i)$ and $z_i \not\doteq z_j$ for all $1 \leq i < j \leq m$. then 1. guess m with $1 \leqslant m \leqslant n$ and set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{(\leqslant mS.C)\}$ 2. create m new nodes $y_1, \ldots, y_m$ with $\mathcal{L}(\langle x, y_i\rangle) = \{S\}$, $\mathcal{L}(y_i) = \{C, o_i\}$ for each $o_i \in N_I$ new in $\mathbf{G}$, and $y_i \not\doteq y_j$ for $1 \leq i < j \leq m$,

Figure 1: The Expansion Rules for the $\mathcal{SROIQ}$ Tableau Algorithm.

As usual, we prove termination, soundness, and completeness of the tableau algorithm to show that it indeed decides satisfiability of $\mathcal{SROIQ}$-concepts w.r.t. Rboxes.

Theorem 19 (Termination, Soundness, and Completeness) *Let C_0 be a $\mathcal{SROIQ}$-concept in NNF and $\mathcal{R}$ a reduced Rbox.*

1. *The tableau algorithm terminates when started with C_0 and $\mathcal{R}$.*
2. *The expansion rules can be applied to C_0 and $\mathcal{R}$ such that they yield a complete and clash-free completion graph if and only if there is a tableau for C_0 w.r.t. $\mathcal{R}$.*

Proof: (1): The algorithm constructs a graph that consists of a set of arbitrarily interconnected nominal nodes, and "trees" of blockable nodes with each tree rooted in r_0 or in a nominal node, and where branches of these trees might end in an edge leading to a nominal node.

Termination is a consequence of the usual $\mathcal{SHIQ}$ conditions with respect to the blockable tree parts of the graph, plus the fact that there is a bound on the number of new nominal nodes that can be added to $\mathbf{G}$ by the *NN*-rule.

The termination proof for the $\mathcal{SROIQ}$ tableaux is virtually identical to the one for $\mathcal{SHOIQ}$, whence we omit the details and refer the reader to (13). To see this, note first that the blocking technique employed for $\mathcal{SROIQ}$ is identical to the one for $\mathcal{SHOIQ}$. Next, the closure $\mathsf{fclos}(C_0, \mathcal{R})$ is defined differently, comprising concepts of the form $\forall \mathcal{B}_S(q).C$, generally yielding a size of $\mathsf{fclos}(C_0, \mathcal{R})$ that can be exponential in the depth of the role hierarchy. However, the construction of the automata can also be considered a pre-prozessing step and part of the input, in that case keeping the polynomial bound on the size of the closure relative to the input. Furthermore, it should be clear that the new Self–Ref-rule (only adding new reflexive edges) as well as the new clash conditions do not affect the termination of the algorithm.

(2): For the "if" direction, we can obtain a tableau $T = (\mathbf{S}, \mathcal{L}', \mathcal{E})$ from a complete and clash-free completion graph $\mathbf{G}$ by *unravelling* blockable "tree" parts of the graph as usual (these are the only parts where blocking can apply).

More precisely, paths are defined as follows. For a label blocked node x, let $b(x)$ denote a node that blocks x.

A **path** is a sequence of pairs of blockable nodes of $\mathbf{G}$ of the form $p = \langle (x_0, x'_0), \ldots, (x_n, x'_n) \rangle$. For such a path, we define $\mathsf{Tail}(p) := x_n$ and $\mathsf{Tail}'(p) := x'_n$. With $\langle p | (x_{n+1}, x'_{n+1}) \rangle$ we denote the path

$$\langle (x_0, x'_0), \ldots, (x_n, x'_n), (x_{n+1}, x'_{n+1}) \rangle.$$

The set $\mathsf{Paths}(\mathbf{G})$ is defined inductively as follows:

- For each blockable node x of $\mathbf{G}$ that is a successor of a nominal node or a root node, $\langle (x, x) \rangle \in \mathsf{Paths}(\mathbf{G})$, and
- For a path $p \in \mathsf{Paths}(\mathbf{G})$ and a blockable node y in $\mathbf{G}$:
  - if y is a successor of $\mathsf{Tail}(p)$ and y is not blocked, then $\langle p | (y, y) \rangle \in \mathsf{Paths}(\mathbf{G})$, and
  - if y is a successor of $\mathsf{Tail}(p)$ and y is blocked, then $\langle p | (b(y), y) \rangle \in \mathsf{Paths}(\mathbf{G})$.

Please note that, due to the construction of Paths, all nodes occurring in a path are blockable and, for $p \in \mathsf{Paths}(\mathbf{G})$ with $p = \langle p' | (x, x') \rangle$, x is not blocked, x' is blocked iff $x \neq x'$, and x' is never indirectly blocked. Furthermore, the blocking condition implies $\mathcal{L}(x) = \mathcal{L}(x')$.

Next, we use $\mathsf{Nom}(\mathbf{G})$ for the set of nominal nodes in $\mathbf{G}$, and define a tableau $T = (\mathbf{S}, \mathcal{L}', \mathcal{E})$ from $\mathbf{G}$ as follows.

$$\mathbf{S} = \mathsf{Nom}(\mathbf{G}) \cup \mathsf{Paths}(\mathbf{G})$$

$$\mathcal{L}'(p) = \begin{cases} \mathcal{L}(\mathsf{Tail}(p)) & \text{if } p \in \mathsf{Paths}(\mathbf{G}) \\ \mathcal{L}(p) & \text{if } p \in \mathsf{Nom}(\mathbf{G}) \end{cases}$$

$$\begin{aligned}\mathcal{E}(R) = \{ \langle p, q \rangle \in \mathsf{Paths}(\mathbf{G}) \times \mathsf{Paths}(\mathbf{G}) \ | \ & \\ & q = \langle p | (x, x') \rangle \text{ and } x' \text{ is an } R\text{-successor of } \mathsf{Tail}(p) \text{ or} \\ & p = \langle q | (x, x') \rangle \text{ and } x' \text{ is an } \mathsf{Inv}(R)\text{-successor of } \mathsf{Tail}(q) \} \cup \\ \{ \langle p, x \rangle \in \mathsf{Paths}(\mathbf{G}) \times \mathsf{Nom}(\mathbf{G}) \ | \ & x \text{ is an } R\text{-neighbour of } \mathsf{Tail}(p) \} \cup \\ \{ \langle x, p \rangle \in \mathsf{Nom}(\mathbf{G}) \times \mathsf{Paths}(\mathbf{G}) \ | \ & \mathsf{Tail}(p) \text{ is an } R\text{-neighbour of } x \} \cup \\ \{ \langle x, y \rangle \in \mathsf{Nom}(\mathbf{G}) \times \mathsf{Nom}(\mathbf{G}) \ | \ & y \text{ is an } R\text{-neighbour of } x \}\end{aligned}$$

We already commented above on $\mathbf{S}$, and $\mathcal{L}'$ is straightforward. Unfortunately, $\mathcal{E}$ is slightly cumbersome because we must distinguish between blockable and nominal nodes.

Claim: T is a tableau for C_0 with respect to $\mathcal{R}$.

Firstly, by definition of the algorithm, there is an heir x_0 of r_0 with $C_0 \in \mathcal{L}(x_0)$. By the $\leqslant$-rule, x_0 is either a root node or a nominal node, and thus cannot be blocked. Hence there is some $s \in \mathbf{S}$ with $C_0 \in \mathcal{L}'(s)$. Next, we prove that T satisfies each (Pi).

- (P1a), (P1b), (P2) and (P3) are trivially implied by the definition of $\mathcal{L}'$ and completeness of $\mathbf{G}$.
- (P1c) follows from the construction of $\mathcal{E}$ and completeness, and (P1d) follows from clash-freeness.
- (P4b) and (P6) follow from completeness of $\mathbf{G}$.
- for (P4a), consider a tuple $\langle s,t\rangle \in \mathcal{E}(R)$ with $\forall\mathcal{B}(p).C \in \mathcal{L}'(s)$ and $p \xrightarrow{R} q \in \mathcal{B}(p)$. We have to show that $\forall\mathcal{B}(q).C \in \mathcal{L}'(t)$ and distinguish four different cases:
  - if $\langle s,t\rangle \in \mathsf{Paths}(\mathbf{G}) \times \mathsf{Paths}(\mathbf{G})$, then $\forall\mathcal{B}(p).C \in \mathcal{L}(\mathsf{Tail}(s))$ and
    - either $\mathsf{Tail}'(t)$ is an R-successor of $\mathsf{Tail}(s)$. Hence completeness implies $\forall\mathcal{B}(q).C \in \mathcal{L}(\mathsf{Tail}'(t))$, and by definition of $\mathsf{Paths}(\mathbf{G})$, either $\mathsf{Tail}'(t) = \mathsf{Tail}(t)$, or $\mathsf{Tail}(t)$ blocks $\mathsf{Tail}'(t)$ and the blocking condition implies $\mathcal{L}(\mathsf{Tail}'(t)) = \mathcal{L}(\mathsf{Tail}(t))$.
    - or $\mathsf{Tail}'(s)$ is an $\mathsf{Inv}(R)$-successor of $\mathsf{Tail}(t)$. Again, either $\mathsf{Tail}'(s) = \mathsf{Tail}(s)$, or $\mathsf{Tail}(s)$ blocks $\mathsf{Tail}'(s)$ in which case the blocking condition implies that $\forall\mathcal{B}(p).C \in \mathcal{L}(\mathsf{Tail}'(s))$, and thus completeness implies that $\forall\mathcal{B}(q).C \in \mathcal{L}(\mathsf{Tail}(t))$.
  - if $\langle s,t\rangle \in \mathsf{Nom}(\mathbf{G}) \times \mathsf{Nom}(\mathbf{G})$, then $\forall\mathcal{B}(p).C \in \mathcal{L}(s)$ and t is an R-neighbour of s. Hence completeness implies $\forall B(q).C \in \mathcal{L}(t)$.
  - if $\langle s,t\rangle \in \mathsf{Nom}(\mathbf{G}) \times \mathsf{Paths}(\mathbf{G})$, then $\forall B(p).C \in \mathcal{L}(s)$ and $\mathsf{Tail}(t)$ is an R-neighbour of s. Hence completeness implies $\forall B(q).C \in \mathcal{L}(\mathsf{Tail}(t))$.
  - if $\langle s,t\rangle \in \mathsf{Paths}(\mathbf{G}) \times \mathsf{Nom}(\mathbf{G})$, then $\forall B(p).C \in \mathcal{L}(\mathsf{Tail}(s))$ and t is an R-neighbour of $\mathsf{Tail}(s)$. Hence completeness implies $\forall B(q).C \in \mathcal{L}(t)$.

  In all four cases, by definition of $\mathcal{L}'$, we have $\forall\mathcal{B}(q).C \in \mathcal{L}'(t)$.

- for (P5), consider some $s \in \mathbf{S}$ with $\exists R.C \in \mathcal{L}'(s)$.
  - If $s \in \mathsf{Paths}(\mathbf{G})$, then $\exists R.C \in \mathcal{L}(\mathsf{Tail}(s))$, $\mathsf{Tail}(s)$ is not blocked, and completeness of $\mathcal{T}$ implies the existence of an R-neighbour y of $\mathsf{Tail}(s)$ with $C \in \mathcal{L}(y)$.
    - If y is a nominal node, then $y \in \mathbf{S}$, $C \in \mathcal{L}'(y)$, and $\langle s, y\rangle \in \mathcal{E}(R)$.
    - If y is blockable and a successor of $\mathsf{Tail}(s)$, then $\langle s|(\tilde{y}, y)\rangle \in \mathbf{S}$, for $\tilde{y} = y$ or $\tilde{y} = b(y)$, $C \in \mathcal{L}'(\langle s|(\tilde{y}, y)\rangle)$, and $\langle s, \langle s|(\tilde{y}, y)\rangle\rangle \in \mathcal{E}(R)$.
    - If y is blockable and a predecessor of $\mathsf{Tail}(s)$, then $s = \langle p|(y, y)|(\mathsf{Tail}(s), \mathsf{Tail}'(s))\rangle$, $C \in \mathcal{L}'(\langle p|(y, y)\rangle)$, and $\langle s, \langle p|(y, y)\rangle\rangle \in \mathcal{E}(R)$.
  - If $s \in \mathsf{Nom}(\mathbf{G})$, then completeness implies the existence of some R-successor x of s with $C \in \mathcal{L}(x)$.
    - If x is a nominal node, then $\langle s, x\rangle \in \mathcal{E}(R)$ and $C \in \mathcal{L}'(x)$.
    - If x is a blockable node, then x is a safe R-neighbour of s and thus not blocked. Hence there is a path $p \in \mathsf{Paths}(\mathbf{G})$ with $\mathsf{Tail}(p) = x$, $\langle s, p\rangle \in \mathcal{E}(R)$ and $C \in \mathcal{L}'(p)$.
- (P7) and (P13) are immediate consequences of the definition of "R-successor" and "R-neighbour", as well as the definition of $\mathcal{E}$.
- for (P8), consider some $s \in \mathbf{S}$ with $(\leqslant nR.C) \in \mathcal{L}'(s)$. Clash-freeness implies the existence of at most n R-neighbours y_i of s with $C \in \mathcal{L}(y_i)$. By construction, each $t \in \mathbf{S}$ with $\langle s, t\rangle \in \mathcal{E}(R)$ corresponds to an R-neighbour y_i of s or $\mathsf{Tail}(s)$, and none of these R-neighbours gives rise to more than one such y_i. Moreover, since $\mathcal{L}'(t) = \mathcal{L}(y_i)$, (P8) is satisfied.
- for (P9), consider some $s \in \mathbf{S}$ with $(\geqslant nR.C) \in \mathcal{L}'(s)$.
  - if $s \in \mathsf{Nom}(\mathbf{G})$, then completeness implies the existence of n safe R-neighbours $y_1, \ldots, y_n$ of s with and $y_j \neq y_j$, for each $i \neq j$, and $C \in \mathcal{L}(y_i)$, for each $1 \leq i \leq n$. By construction, each y_i corresponds to a $t_i \in \mathbf{S}$ with $t_i \neq t_j$, for each $i \neq j$:
    - if y_i is blockable, then it cannot be blocked since it is a safe R-neighbour of s. Hence there is a path $\langle p|(y_i, y_i)\rangle \in \mathbf{S}$ and $\langle s, \langle p|(y_i, y_i)\rangle\rangle \in \mathcal{E}(R)$.

  - $*$ if y_i is a nominal node, then $\langle s, y_i \rangle \in \mathcal{E}(R)$.

- if $s \in \mathsf{Paths}(\mathbf{G})$, then completeness implies the existence of n R-neighbours $y_1, \ldots, y_n$ of $\mathsf{Tail}(s)$ with $y_j \neq y_j$, for each $i \neq j$, and $C \in \mathcal{L}(y_i)$, for each $1 \leq i \leq n$. By construction, each y_i corresponds to a $t_i \in \mathbf{S}$ with $t_i \neq t_j$, for each $i \neq j$:
  - $*$ if y_i is safe, then it can be blocked if it is a successor of $\mathsf{Tail}(s)$. In this case, the "pair" construction in our definition of paths ensure that, even if $b(y_i) = b(y_j)$, for some $i \neq j$, we still have $\langle p|(b(y_i), y_i)\rangle \neq \langle p|(b(y_j), b_j)\rangle$.
  - $*$ if y_i is unsafe, then $\langle s, y_i \rangle \in \mathcal{E}(R)$.

  Hence all t_i are different and, by construction, $C \in \mathcal{L}'(t_i)$, for each $1 \leq i \leq n$.

- (P10) is satisfied due to completeness of $\mathbf{G}$ and the fact that each $t \in \mathbf{S}$ with $\langle s, t \rangle \in \mathcal{E}(R)$ corresponds to an R-neighbour of s (in case $s \in \mathsf{Nom}(\mathbf{G})$) or of $\mathsf{Tail}(s)$ (in case $s \in \mathsf{Paths}(\mathbf{G})$).
- (P11) follows from clash-freeness and definition of $\mathcal{E}$.
- (P12) follows from completeness of $\mathbf{G}$ and definition of $\mathcal{E}$ (just as (P1c)).
- (P14a) follows trivially from the initialisation of $\mathbf{G}$.
- (P14b) is due to completeness of $\mathbf{G}$ and the fact that nominal nodes are not "unravelled".

For the "only if" direction, given a tableau $T = (\mathbf{S}, \mathcal{L}', \mathcal{E})$ for C_0 w.r.t. $\mathcal{R}$, we can apply the non-deterministic rules, i.e., the $\sqcup$-, *choose*-, $\leqslant$-, and *NN*-rule, in such a way that we obtain a complete and clash-free graph: inductively with the generation of new nodes, we define a mapping π from nodes in the completion graph to individuals in $\mathbf{S}$ of the tableau in such a way that,

1. for each node x, $\mathcal{L}(x) \subseteq \mathcal{L}'(\pi(x))$,
2. for each pair of nodes x, y and each role R, if y is an R-successor of x, then $\langle \pi(x), \pi(y) \rangle \in \mathcal{E}(R)$, and

3. $x \dot{\neq} y$ implies $\pi(x) \neq \pi(y)$.

This is analogous to the proof in (15) with the additional observation that, due to (P14b), application of the o-rule does not lead to a clash of the form (6) as given in Definition 18. Similarly, an application of the Self–Ref-rule does not lead to a clash of the form (3) due to Conditons (P1d), and a clash of the form (4) can not occur due to (P11).

□

From Theorems 9, 17 and 19, we thus arrive at the following theorem:

Theorem 20 (Decidability) *The tableau algorithm decides satisfiability and subsumption of $\mathcal{SROIQ}$-concepts with respect to Aboxes, Rboxes, and Tboxes.*

References

[1] Areces, C., Blackburn, P., Hernandez, B., and Marx, M. Handling Boolean Aboxes. In *Proc. of the 2003 Description Logic Workshop (DL 2003)* (2003), CEUR (`http://ceur-ws.org/`).

[2] Baader, F. Augmenting Concept Languages by Transitive Closure of Roles: An Alternative to Terminological Cycles. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)* (Sydney, 1991).

[3] Baader, F., Bürckert, H.-J., Nebel, B., Nutt, W., and Smolka, G. On the Expressivity of Feature Logics with Negation, Functional Uncertainty, and Sort Equations. *Journal of Logic, Language and Information 2* (1993), 1–18.

[4] Baader, F., and Hanschke, P. A schema for integrating concrete domains into concept languages. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)* (1991), pp. 452–457.

[5] Baader, F., Lutz, C., Milicic, M., Sattler, U., and Wolter, F. Integrating Description Logics and Action Formalisms: First Results. In *Proc. of the 20th National Conference on Artificial Intelligence (AAAI-05)* (2005), A. Press, Ed.

[6] Blackburn, P., and Seligman, J. Hybrid languages. *J. of Logic, Language and Information 4* (1995), 251–272.

[7] Hopcroft, J. E., and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley Publ. Co., Reading, Massachussetts, 1997.

[8] Horrocks, I., Kutz, O., and Sattler, U. The Irresistible $\mathcal{SRIQ}$. In *Proc. of Workshop: OWL: Experiences and Directions, Galway, Ireland, November 11th-12th* (2005).

[9] Horrocks, I., Patel-Schneider, P. F., and van Harmelen, F. From $\mathcal{SHIQ}$ and RDF to OWL: The Making of a Web Ontology Language. *J. of Web Semantics 1*, 1 (2003), 7–26.

[10] Horrocks, I., and Sattler, U. Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)* (2001), pp. 199–204.

[11] Horrocks, I., and Sattler, U. Optimised reasoning for $\mathcal{SHIQ}$. In *Proc. of the 15th European Conf. on Artificial Intelligence (ECAI 2002)* (2002).

[12] Horrocks, I., and Sattler, U. Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms. *Artificial Intelligence 160* (2004), 79–104.

[13] Horrocks, I., and Sattler, U. A Tableaux Decision Procedure for $\mathcal{SHOIQ}$. In *Proc. of 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)* (2005), Morgan Kaufmann, Los Altos.

[14] Horrocks, I., Sattler, U., and Tobies, S. Practical Reasoning for Expressive Description Logics. In *Proc. of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99)* (1999), H. Ganzinger, D. McAllester, and A. Voronkov, Eds., vol. 1705 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, pp. 161–180.

[15] Horrocks, I., Sattler, U., and Tobies, S. Practical reasoning for expressive description logics. In *Proc. of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99)* (1999), H. Ganzinger, D. McAllester, and A. Voronkov, Eds., no. 1705 in Lecture Notes in Artificial Intelligence, Springer, pp. 161–180.

[16] Horrocks, I., Sattler, U., and Tobies, S. Reasoning with individuals for the description logic $\mathcal{SHIQ}$. In *Proc. of the 17th Int. Conf. on Automated Deduction (CADE 2000)* (2000), D. McAllester, Ed., vol. 1831 of *Lecture Notes in Computer Science*, Springer, pp. 482–496.

[17] Horrocks, I., Sattler, U., and Tobies, S. Reasoning with individuals for the description logic $\mathcal{SHIQ}$. In *Proc. of the 17th Conf. on Automated Deduction (CADE-17)* (Germany, 2000), D. MacAllester, Ed., vol. 1831 of *Lecture Notes in Computer Science*, Springer-Verlag.

[18] Pan, J., and Horrocks, I. Web ontology reasoning with datatype groups. In *Proc. of the 2003 International Semantic Web Conference (ISWC 2003)* (2003), D. Fensel, K. Sycara, and J. Mylopoulos, Eds., no. 2870 in Lecture Notes in Computer Science, Springer, pp. 47–63.

[19] Schaerf, A. Reasoning with individuals in concept languages. *Data and Knowledge Engineering 13*, 2 (1994), 141–176.

[20] Schild, K. A Correspondence Theory for Terminological Logics: Preliminary Report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)* (Sydney, 1991), pp. 466–471.

[21] Wolstencroft, K., Brass, A., Horrocks, I., Lord, P., Sattler, U., Turi, D., and Stevens, R. A Little Semantic Web Goes a Long Way in Biology. In *Proc. of the 4th International Semantic Web Conference* (2005), LNCS, SV. To appear.