The Irresistible $\mathcal{SRIQ}$

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. Motivated primarily by medical terminology applications, the prominent DL $\mathcal{SHIQ}$ has already been extended to a DL with complex role inclusion axioms of the form $R \circ S \,\dot{\sqsubseteq}\, R$ or $S \circ R \,\dot{\sqsubseteq}\, R$, called $\mathcal{RIQ}$, and the $\mathcal{SHIQ}$ tableau algorithm has been extended to handle such inclusions.
This paper further extends $\mathcal{RIQ}$ and its tableau algorithm with important expressive means that are frequently requested in ontology applications, namely with reflexive, symmetric, transitive, and irreflexive roles, disjoint roles, and the construct $\exists R.\mathsf{Self}$, allowing, for instance, the definition of concepts such as a "narcist". Furthermore, we extend the algorithm to cover Abox reasoning extended with negated role assertions. The resulting logic is called $\mathcal{SRIQ}$.

1 Introduction

We describe an extension, called $\mathcal{SRIQ}$, of the description logic (DL) $\mathcal{SHIN}$ (10) underlying OWL lite and OWL DL (7). We believe that $\mathcal{SRIQ}$ enjoys some useful properties. Firstly, $\mathcal{SRIQ}$ extends $\mathcal{SHIN}$ with numerous expressive means which have been asked for by users, and which, we believe, will make modeling using DLs easier and more intuitive. While the language of $\mathcal{SRIQ}$ is designed to be slightly redundant in the sense that some of the new expressive means can be simulated by others, the complete absence of those expressive means has proven quite harmful since developers of ontologies use work-arounds to compensate for this. As a consequence, ontologies become cluttered, complicated, and difficult to understand. In the worst case, the work-around only partially captures the intended semantics, thus leading to unintended or missing consequences, thereby destroying one of the main features of a logic-based formalism, namely its well-defined semantics and reasoning services. A well-known example are *qualified* number restrictions. Their absence in OWL lite and OWL DL has caused problems in the past (12), and has led to the development and use of questionable surrogates. Hence, $\mathcal{SRIQ}$ provides qualified number restrictions. Other, novel expressive means of $\mathcal{SRIQ}$ concern mostly roles and include:

– *disjoint roles.* E.g., the roles `sister` and `mother` could be declared as being disjoint. Most DLs can be said to be "lopsided" 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.

– *reflexive and irreflexive roles*. E.g., the role `knows` could be declared as being reflexive, and the role `sibling` could be declared as being irreflexive. In the presence of the new concept $\exists R.\mathsf{Self}$ described below, reflexive and irreflexive roles also become definable by Tbox assertions.
– *negated role assertions*. Most Abox formalisms only allow for positive role assertions (with few exceptions (1; 5)), whereas $\mathcal{SRIQ}$ also allows for statements such as $(\mathtt{John}, \mathtt{Mary}) : \neg\mathtt{likes}$. In the presence of complex role inclusions, negated role assertions can be quite useful and, like disjoint roles, they overcome a certain "lopsidedness" of DLs.
– Since $\mathcal{SRIQ}$ extends $\mathcal{SHIQ}$, we can also express that a role is transitive or symmetric, and can use role inclusion axioms $R \sqsubseteq S$.
– Since $\mathcal{SRIQ}$ extends $\mathcal{RIQ}$ (8), we can use complex role inclusion axioms of the form $R \circ S \dot{\sqsubseteq} R$ and $S \circ R \dot{\sqsubseteq} R$. 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: $\exists \mathtt{owns}.\mathtt{Car} \sqsubseteq \exists \mathtt{owns}.\mathtt{Engine}$.
– Finally, $\mathcal{SRIQ}$ 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" using $\exists \mathtt{likes}.\mathsf{Self}$.

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

Secondly, $\mathcal{SRIQ}$ is designed to be of similar practicability as $\mathcal{SHIQ}$. The tableau algorithm for $\mathcal{SHIQ}$ and the one for $\mathcal{SRIQ}$ presented here are very similar. Even though the additional expressive means of $\mathcal{SRIQ}$ require certain adjustments to the $\mathcal{SHIQ}$ algorithm, 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 $\mathcal{SHIQ}$ algorithm. More precisely, we employ the same technique using finite automata as in (8) 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.

The current paper describes work in progress towards a description logic that overcomes certain shortcomings in expressiveness of other DLs. We have used $\mathcal{SHIN}$, $\mathcal{SHIQ}$, and $\mathcal{RIQ}$ as a starting point, extended them with some "useful-yet-harmless" expressive means, and also extended the tableau algorithm accordingly. We wish to discuss this extension in case we have overlooked other "useful-yet-harmless" expressive means, and we plan to further extend $\mathcal{SRIQ}$: currently, various new operators are restricted to *simple* roles, and we have yet to establish which of these restrictions are necessary in order to preserve decidability[1] or practicability. Moreover, we plan to extend $\mathcal{SRIQ}$ towards $\mathcal{SHOIQ}$ (9), i.e., to also include nominals.

For a full specification of the tableau algorithm and proofs, see (6).

2 The Logic $\mathcal{SRIQ}$

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

2.1 Roles, Role Hierarchies, and Role Assertions

Definition 1 (Interpretations). *Let* $\mathbf{C}$ *be a set of* ***concept names****,* $\mathbf{R}$ *a set of* ***role names****, and* $\mathbf{I} = \{a, b, c \dots\}$ *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 each concept name* C *a subset* $C^\mathcal{I} \subseteq \Delta^\mathcal{I}$ *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*

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

Note that, unlike in the case of $\mathcal{SHIQ}$, we did not introduce *transitive role names*. This is so since, 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 $w = R_1 \dots R_n$ for R_i roles, then we set $w^\mathcal{I} = R_1^\mathcal{I} \circ \dots \circ R_n^\mathcal{I}$ and $\mathsf{Inv}(w) = \mathsf{Inv}(R_n) \dots \mathsf{Inv}(R_1)$, 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*, i.e., statements of the form $R \mathrel{\dot\sqsubseteq} S$, $RS \mathrel{\dot\sqsubseteq} S$, and $SR \mathrel{\dot\sqsubseteq} S$. The second component is a set $\mathcal{R}_a$ of *role assertions* stating, for instance, that a role R must be interpreted as a transitive, reflexive, irreflexive, symmetric, or transitive 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 role hierarchy, whose definition involves a strict partial order $\prec$ on roles, i.e., an irreflexive and transitive relation on $R \cup \{R^- \mid R \in \mathbf{R}\}$.

Definition 2 ((Regular) Role Inclusion Axioms).

Let $\prec$ *be a strict partial order on roles. A* ***role inclusion axiom*** *(RIA for short) is an expression of the form* $w \mathrel{\dot\sqsubseteq} R$*, 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.*

[1] See (10) for such a case.

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

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

*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,*
- *$w = RR$,*
- *$w = R^-$,*
- *$w = S_1 \dots S_n$ and $S_i \prec R$, for all $1 \leq i \leq n$,*
- *$w = RS_1 \dots S_n$ and $S_i \prec R$, for all $1 \leq i \leq n$, or*
- *$w = S_1 \dots S_n R$ 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 strict partial 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

$$\{RS \dot{\sqsubseteq} S, \quad RT \dot{\sqsubseteq} R, \quad UT \dot{\sqsubseteq} T, \quad US \dot{\sqsubseteq} U\}$$

is not regular because it would require $\prec$ to satisfy $S \prec U \prec T \prec R \prec S$, which would imply $S \prec S$, thus contradicting irreflexivity. Such cyclic dependencies are known to lead to undecidability (8).

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

Note that, due to the fourth restriction in the definition of $\prec$-regularity, we also restrict $\underline{\circledast}$ 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 $\{\langle x, x \rangle \mid x \in \Delta^{\mathcal{I}}\}$ and set $R^{\mathcal{I}} \downarrow := \{\langle x, x \rangle \mid \exists y \in \Delta^{\mathcal{I}}. \langle x, y \rangle \in R^{\mathcal{I}}\}$.

[2] This is not a serious restriction for, if $\mathcal{R}$ contains $\underline{\circledast}$ 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 and S, 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:*

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

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 symmetric roles will be handled by the RIAs alone.

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 $\mathcal{SHIQ}$, the application of qualified number restrictions has to be restricted to certain roles, called *simple roles*, to preserve decidability (10). In the context of $\mathcal{SRIQ}$, 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 $\mathsf{Ref}(R)$, $\mathsf{Irr}(R)$, 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 $\mathcal{SRIQ}$-**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 $\mathcal{SRIQ}$

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

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

- *every concept name 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 $\mathcal{SRIQ}$-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 $a \not\doteq b$, for $a, b \in \mathbf{I}$ (the set of individual names), a (possibly inverse) role R, a (possibly inverse) simple role S, and a $\mathcal{SRIQ}$-concept C. A $\mathcal{SRIQ}$-* ***Abox*** *$\mathcal{A}$ is a finite set of individual assertions.*

Note that number restrictions $(\geqslant nS.C)$ and $(\leqslant nS.C)$, the concept $\exists S.\mathsf{Self}$, and negated role assertions $(a, b) : \neg S$, are all restricted to *simple* roles. In the case of number restrictions we mentioned the reason for this restriction already: without it, already the satisfiability problem of $\mathcal{SHIQ}$-concepts is undecidable (10), 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 $\mathcal{SRIQ}$ 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 are necessary in order to preserve decidability or practicability. For example, it should be possible to also allow non-simple roles in negated role assertions $(a, b) : \neg R$ without losing decidability.

Note also that, in the definition of $\mathcal{SRIQ}$-Aboxes, we do not assume the unique name assumption (UNA) (which is commonly assumed in DLs (4)). Rather, by allowing inequalities between individuals in the Abox to be explicitly stated, we increase flexibility while, obviously, the UNA can be regained by explicitly stating $a \not\doteq b$ for every pair $a, b \in \mathbf{I}$ of individuals. Moreover, notice that, in contrast to standard Aboxes, $\mathcal{SRIQ}$-Aboxes can also contain negated role assertions of the form $(a, b) : \neg R$.

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:*

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

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

A concept C is called ***satisfiable*** *iff there is an interpretation $\mathcal{I}$ with $C^{\mathcal{I}} \neq \emptyset$. A concept D* ***subsumes*** *a concept C (written $C \sqsubseteq D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ 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 iff $x \in C^{\mathcal{I}}$.*

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*

$$\begin{array}{llll} \mathcal{I} \models a{:}C & \text{if } a^{\mathcal{I}} \in C^{\mathcal{I}}; & \mathcal{I} \models a \not\doteq b & \text{if } a^{\mathcal{I}} \neq b^{\mathcal{I}}; \\ \mathcal{I} \models (a,b){:}R & \text{if } \langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}; & \mathcal{I} \models (a,b){:}\neg R & \text{if } \langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \notin R^{\mathcal{I}}. \end{array}$$

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 $\mathcal{I} \models \mathcal{A}$.*

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 $\{a{:}C\}$ 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; 11; 3), the *internalisation* of GCIs is introduced, a technique that realises exactly this reduction. For $\mathcal{SRIQ}$, this technique can be modified accordingly.

Now, note also that, instead of having a role assertion $\mathsf{Ref}(R) \in \mathcal{R}_a$, we can add, equivalently, the GCI $\exists R.\top \dot{\sqsubseteq} \exists R.\mathsf{Self}$ to $\mathcal{T}$, which can in turn be internalised. Likewise, instead of asserting $\mathsf{Irr}(R)$, we can, equivalently, add the GCI $\top \dot{\sqsubseteq} \neg\exists R.\mathsf{Self}$. Thus, we arrive at the following theorem:

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

2. Consistency of $\mathcal{SRIQ}$-Aboxes w.r.t. Tboxes and Rboxes is polynomially reducible to consistency of $\mathcal{SRIQ}$-Aboxes w.r.t. Rboxes.

3. W.l.o.g., we can assume that Rboxes do not contain role assertions of the form $\mathsf{Irr}(R)$, $\mathsf{Ref}(R)$, $\mathsf{Tra}(R)$, or $\mathsf{Sym}(R)$.

With Theorem 1, all standard inference problems for $\mathcal{SRIQ}$-concepts and Aboxes can be reduced to the problem of determining the consistency of a $\mathcal{SRIQ}$-Abox w.r.t. to an Rbox, where we can assume w.l.o.g. that all role assertions in the Rbox are of the form $\mathsf{Dis}(R, S)$—we call such an Rbox **reduced**.

3 $\mathcal{SRIQ}$ is Decidable

As we have just seen, we can restrict our attention to the consistency of Aboxes w.r.t. reduced Rboxes only. We have extended the tableau algorithm for $\mathcal{RIQ}$ to $\mathcal{SRIQ}$, and will spend the remainder of this paper on its description.

In a first step, the tableau algorithm takes a reduced Rbox $\mathcal{R}$ and an Abox $\mathcal{A}$ and builds, for each possibly inverse role R occurring in $\mathcal{R}$ or $\mathcal{A}$, a non-deterministic *finite automaton* $\mathcal{B}_R$. Intuitively, such 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. The following proposition states that $\mathcal{B}_R$ indeed captures all implications between (paths of) roles and R that are consequences of the role hierarchy $\mathcal{R}_h$, where $L(\mathcal{B}_R)$ denotes the language (a set of strings of roles) accepted by $\mathcal{B}_R$.

Proposition 1. *$\mathcal{I}$ is a model of $\mathcal{R}_h$ if and only if, for each (possibly inverse) role R occurring in $\mathcal{R}_h$, each word $w \in L(\mathcal{B}_R)$, and each $\langle x, y\rangle \in w^{\mathcal{I}}$, we have $\langle x, y\rangle \in R^{\mathcal{I}}$.*

Since Aboxes usually involve several individuals with arbitrary role relationships between them, the completion algorithm presented works on *forests* rather than on *trees*. A forest is a collection of trees whose root nodes correspond to the individuals appearing in the input Abox and which form an arbitrarily connected graph according to the role assertions stated in the Abox. Similar as for $\mathcal{RIQ}$, we define a set $\mathsf{fclos}(\mathcal{A}, \mathcal{R})$ of "relevant sub-concepts" of those concepts occurring in $\mathcal{A}$; see (6) for details.

Definition 7. *A **completion forest** $\mathbf{F}$ for a $\mathcal{SRIQ}$-Abox $\mathcal{A}$ and an Rbox $\mathcal{R}$ is a collection of trees whose distinguished **root nodes** can be connected arbitrarily. Moreover, each node x is labelled with a set $\mathcal{L}(x) \subseteq \mathsf{fclos}(\mathcal{A}, \mathcal{R})$ and each edge $\langle x, y\rangle$ from a node x to its **successor** y is labelled with a non-empty set $\mathcal{L}(\langle x, y\rangle)$ of (possibly inverse and possibly negated) roles occurring in $\mathcal{A}$ and $\mathcal{R}$. Finally, completion forests come with an explicit **inequality relation** $\neq$ on nodes which is implicitly assumed to be symmetric.*

*Let x and y be nodes in $\mathbf{F}$ and R a role. If $R' \mathrel{\underline{\circledast}} R$ and $R' \in \mathcal{L}(\langle x, y\rangle)$, then y is called an R-**successor** of x.*

*If y is an R-successor of x or x is an $\mathsf{Inv}(R)$-successor of y, then y is called an R-**neighbour** of x. Moreover, a node x is a **neighbour** of y, if it is an R-neighbour for some role R. **Successors**, **predecessors**, **ancestors**, and **descendants** are defined as usual.*

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

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

*A node is **blocked** iff 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. *none of x', y and y' is a root node,*

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 say that y ***blocks*** x.

A node y *is* ***indirectly blocked*** *if one of its ancestors is blocked.*

Given a non-empty $\mathcal{SRIQ}$*-Abox* $\mathcal{A}$ *and a reduced Rbox* $\mathcal{R}$*, the tableau algorithm is initialised with the completion forest* $\mathbf{F}_{\mathcal{A},\mathcal{R}}$ *defined as follows:*

- *for each individual* a *occurring in* $\mathcal{A}$, $\mathbf{F}_{\mathcal{A},\mathcal{R}}$ *contains a* ***root node*** x_a,
- *if* $(a, b){:}R \in \mathcal{A}$ *or* $(a, b){:}\neg R \in \mathcal{A}$, *then* $\mathbf{F}_{\mathcal{A},\mathcal{R}}$ *contains an edge* $\langle x_a, x_b\rangle$,
- *if* $a \neq b \in \mathcal{A}$, *then* $x_a \neq x_b$ *is in* $\mathbf{F}_{\mathcal{A},\mathcal{R}}$,
- $\mathcal{L}(x_a) := \{C \mid a{:}C \in \mathcal{A}\}$, *and*
- $\mathcal{L}(\langle x_a, x_b\rangle) := \{R \mid (a, b){:}R \in \mathcal{A}\} \cup \{\neg R \mid (a, b){:}\neg R \in \mathcal{A}\}$.

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

1. $\bot \in \mathcal{L}(x)$, *or*
2. *for some concept name* A, $\{A, \neg A\} \subseteq \mathcal{L}(x)$, *or*
3. x *is an* S*-neighbour of* x *and* $\neg\exists S.\mathsf{Self} \in \mathcal{L}(x)$, *or*
4. x *and* y *are root nodes,* y *is an* R*-neighbour of* x*, and* $\neg R \in \mathcal{L}(\langle x, y\rangle)$, *or*
5. *there is some* $\mathsf{Dis}(R, S) \in \mathcal{R}_a$ *and* y *is an* R*- and an* S*-neighbour of* x*, or*
6. *there is some concept* $(\leqslant nS.C) \in \mathcal{L}(x)$ *and* $\{y_0, \ldots, y_n\} \subseteq S^{\mathbf{F}}(x, C)$ *with* $y_i \not\doteq y_j$ *for all* $0 \leq i < j \leq n$.

A completion forest that does not contain a clash is called ***clash-free****. A completion forest is* ***complete*** *if none of the rules from Figure 1 can be applied to it.*

When started with a non-empty Abox $\mathcal{A}$ *and a reduced Rbox* $\mathcal{R}$*, the tableau algorithm initialises* $\mathbf{F}_{\mathcal{A},\mathcal{R}}$ *and repeatedly applies the expansion rules from Figure 1 to it, stopping when a clash occurs, and applying the shrinking rules eagerly, i.e., the* $\leq$*- and the* $\leq_r$*-rule are applied with highest priority. The algorithm answers "*$\mathcal{A}$ *is satisfiable w.r.t.* $\mathcal{R}$*" iff the expansion rules can be applied in such a way that they yield a complete and clash-free completion forest, and "*$\mathcal{A}$ *is unsatisfiable w.r.t.* $\mathcal{R}$*" otherwise.*

Lemma 1. *Let* $\mathcal{A}$ *be a* $\mathcal{SRIQ}$*-Abox where all concepts are in negation normal form and* $\mathcal{R}$ *a reduced Rbox.*

- *The tableau algorithm terminates when started for* $\mathcal{A}$ *and* $\mathcal{R}$.
- *The expansion rules can be applied to* $\mathcal{A}$ *and* $\mathcal{R}$ *such that they yield a complete and clash-free completion forest iff there is a tableau for* $\mathcal{A}$ *w.r.t.* $\mathcal{R}$.

From Theorem 1 and Lemma 1, we thus have the following theorem:

Theorem 2. *The tableau algorithm decides satisfiability and subsumption of* $\mathcal{SRIQ}$*-concepts with respect to Aboxes, Rboxes, and Tboxes.*

$\sqcap$-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-rule: if $\exists S.\mathsf{Self} \in \mathcal{L}(x)$, 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 $(\geqslant nS.C) \in \mathcal{L}(x)$, x is not blocked, and
there are no $y_1, \ldots, y_n \in S^{\mathbf{F}}(x, C)$
with $y_i \neq y_j$ for each $1 \leq i < j \leq n$
then create n new successors $y_1, \ldots, y_n$ of x 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 $(\leqslant nS.C) \in \mathcal{L}(x)$, x is not indirectly blocked, and
$\# S^{\mathbf{F}}(x, C) > n$, there are $y, z \in S^{\mathbf{F}}(x, C)$ with
not $y \neq z$ and y is not a root node nor 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. Set $u \neq z$ for all u with $u \neq y$.
4. remove y and the sub-tree below y from $\mathbf{F}$.

$\leqslant_r$-rule: if $(\leqslant nS.C) \in \mathcal{L}(x)$, $\# S^{\mathbf{F}}(x, C) > n$,
and there are two root nodes $y, z \in S^{\mathbf{F}}(x, C)$ with *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 $\mathcal{L}(\langle z, w\rangle) := \mathcal{L}(\langle y, w\rangle)$;
ii. else $\mathcal{L}(\langle z, w\rangle) \longrightarrow \mathcal{L}(\langle z, w\rangle) \cup \mathcal{L}(\langle y, w\rangle)$.
3. For all edges $\langle w, y\rangle$:
i. if the edge $\langle w, z\rangle$ does not exist, create it with $\mathcal{L}(\langle w, z\rangle) := \mathcal{L}(\langle w, y\rangle)$;
ii. else $\mathcal{L}(\langle w, z\rangle) \longrightarrow \mathcal{L}(\langle w, z\rangle) \cup \mathcal{L}(\langle w, y\rangle)$.
4. Set $u \neq z$ for all u with $u \neq y$.
5. Remove y and all incoming and outgoing edges from y from $\mathbf{F}$.

Fig. 1. The Expansion Rules for the $\mathcal{SRIQ}$ Tableau Algorithm.

Bibliography

[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., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. F., Eds. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[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] Horrocks, I., Kutz, O., and Sattler, U. The Irresistible $\mathcal{SRIQ}$. Tech. rep., University of Manchester, 2005. Available at `http://www.cs.man.ac.uk/~sattler/publications/sriq-tr.pdf`.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.