An Abstract Tableau Calculus for the Description Logic $\mathcal{SHOI}$ Using Unrestricted Blocking and Rewriting

Mohammad Khodadadi, Renate A. Schmidt, and Dmitry Tishkovsky[⋆]

School of Computer Science, The University of Manchester, UK

Abstract This paper presents an abstract tableau calculus for the description logic $\mathcal{SHOI}$. $\mathcal{SHOI}$ is the extension of $\mathcal{ALC}$ with singleton concepts, role inverse, transitive roles and role inclusion axioms. The presented tableau calculus is inspired by a recently introduced tableau synthesis framework. Termination is achieved by a variation of the unrestricted blocking mechanism that immediately rewrites terms with respect to the conjectured equalities. This approach leads to reduced search space for decision procedures based on the calculus. We also discuss restrictions of the application of the blocking rule by means of additional side conditions and/or additional premises.

1 Introduction

Since the late nineteen eighties various tableau algorithms have been developed for description logics [2]. The way they are defined and blocking is performed these tableau algorithms exploit in an essential way that the supported description logics have a kind of tree-model property. The basic idea is to perform the derivations such that tree-like models are constructed by systematically creating maximally expanded label sets of concept expressions for individual terms one-by-one in a stratified way. Blocking can then be used to ensure no two individuals (perhaps, in an ancestor relationship) have the same label sets, or are a subset of other label sets. For description logics with role inverse, nominals and number restrictions, this kind of stratified construction is more complex requiring some back-and-forth traversal of a tree model together with forms of dynamic blocking [9,10]. This more complex non-local construction is still aimed at finding tree models and therefore not sufficient for description logics without a kind of tree-model property.

In [14,13] we show description logics without the tree-model property, in particular, the description logics $\mathcal{ALBO}$ and $\mathcal{ALBO}^{\mathsf{id}}$, can be decided using a labelled tableau approach enhanced with the so-called unrestricted blocking mechanism. Labelled tableau approaches are common for modal logics, hybrid logics and various other non-classical logics, cf. e.g., [5,3,4,1]. Labelled tableau approaches are easy to understand, they are easy to define as abstract calculi,

[⋆] This research is supported by EPSRC research grant EP/H043748/1.

even for undecidable logics, and are not limited to logics with a form of tree model property. There is also more flexibility in the way that derivations can be performed and it is thus easy to devise sound and complete tableau calculi. Building on [14,13] we have devised a framework for systematically developing labelled tableau calculi for various logics, not only description logics [12]. Essentially for any logic whose semantic definition can be specified in the specification language of the framework, a sound and complete labelled tableau calculus can be synthesised, if certain general conditions hold.

Being based on a sound tableau rule and equality reasoning, the unrestricted blocking mechanism is generally sound and can be incorporated into sound and complete labelled tableau calculi or related approaches. We have shown that if the logic has the finite model property then adding the unrestricted blocking mechanism guarantees also termination [11,12]. Unrestricted blocking provides an intuitively simple method for obtaining termination and behaves very different to standard blocking techniques. It does not require specialised blocking tests and complicated dynamic processing steps. All individuals are blockable and once blocked remain blocked. It can be used to find small finite models.

The aim of this paper is to formalise reasoning for a well-studied, expressive description logic in an abstract labelled tableau calculus incorporating unrestricted blocking. We also we want to explore the possibilities of emulating different kinds of existing blocking techniques. In particular, we present an abstract labelled tableau calculus for the description logic $\mathcal{SHOI}$. The tableau calculus is in line with a refined tableau calculus obtained in the tableau synthesis framework, but exploiting the tree model property of $\mathcal{SHOI}$, transitive roles are accommodated via a propagation rule rather than a structural rule.

Different to most labelled tableau approaches the expansion of $\exists$ expressions introduces Skolem terms rather than constants. Another novelty is the use of ordered rewriting to realise equality reasoning for singleton concepts (nominals) and blocking. Though there are similarities with substitution and nominal deletion approaches (e.g., [10,4,1]), using Skolem terms and ordered rewriting avoids the need to perform again some inference steps on the same branch. Also significantly fewer inferences are performed than when using standard tableau rules for equality as in, e.g., [3,14,13]. As the unrestricted blocking rule is generally sound, any restriction of the rule obtained by adding side-conditions or premises is also sound. This makes it possible to restrict the application of blocking without losing soundness and completeness. For example, it is possible to approximate standard loop checking techniques such as subset ancestor blocking or anywhere equality blocking and simulate approaches using the δ^*-rule.

The paper is structured as follows. The syntax and semantics of $\mathcal{SHOI}$ are defined in Section 2. In Section 3 we define the tableau calculus for $\mathcal{SHOI}$ and in Section 4 we prove that it is sound, complete and terminating. Furthermore, we present examples of restricting the blocking rule by imposing constraints via additional premises and/or side conditions in Section 4. Due to space restrictions we do not discuss the emulation of all known existing blocking techniques but the examples given illustrate the general idea.

2 The Description Logic $\mathcal{SHOI}$

$\mathcal{SHOI}$ extends the description logic $\mathcal{ALC}$ with singleton concepts, role inverse, transitive roles and role inclusion axioms. The language of $\mathcal{SHOI}$ is defined over disjoint countable sets of concept names (atomic concepts), individuals, and role names (atomic roles). The symbol A is used to denote an atomic concept, the symbols a and b denote individuals, and the symbol r denotes an atomic role. Concept and role expressions are built from atomic concepts, individuals, and atomic roles using connectives $\{\cdot\}$ (singleton operator), $\neg$, $\sqcup$, and $\exists\cdot.\cdot$ (existential restriction operator), $^-$ (role inverse operator). Formally, concept and role expressions are defined respectively by the following grammar rules, where C and D denote concept expressions and R denotes a role expression.

$$\begin{aligned} C, D &\overset{\mathsf{def}}{=} A \mid \{a\} \mid \neg C \mid C \sqcup D \mid \exists R.C \\ R &\overset{\mathsf{def}}{=} r \mid R^- \end{aligned}$$

The operators $\top$, $\bot$, $\sqcap$, and $\forall\cdot.\cdot$ are defined as usual. In order to simplify the syntax and avoid repetitive occurrences of the role inverse operator we assume that $(r^-)^- \overset{\mathsf{def}}{=} r$. Further, in $\mathcal{SHOI}$, any atomic role is allowed to be declared as transitive and the predicate Trans is used to denote this. Thus, for every atomic role r, $\mathsf{Trans}(r)$ is true iff r is transitive.

A description logic knowledge base consists of an ABox $\mathcal{A}$, a TBox $\mathcal{T}$ and an RBox $\mathcal{R}$. The ABox consists of a finite number of concept assertions of the form $a : C$ and role assertions of the form $(a, b) : R$. The TBox is used to express a hierarchy between concepts through a finite set of inclusion statements of the form $C \sqsubseteq D$. A *normalised* TBox is a set of inclusion statements of the form $\top \sqsubseteq C$. The RBox is a finite set of inclusion statements of the form $R \sqsubseteq S$ and $\mathsf{Trans}(r)$, to specify a hierarchy between roles and define transitivity of some roles. We define the *closure* $\mathcal{R}^+$ of the RBox $\mathcal{R}$ as the smallest RBox that contains $\mathcal{R}$ and satisfies the following properties.

- if $Q \sqsubseteq R \in \mathcal{R}^+$ then $Q^- \sqsubseteq R^- \in \mathcal{R}^+$;
- if $Q \sqsubseteq R, R \sqsubseteq S \in \mathcal{R}^+$ then $Q \sqsubseteq S \in \mathcal{R}^+$.

Given an RBox $\mathcal{R}$, let $\mathcal{R}^*$ denote the RBox $\mathcal{R}^+ \cup \{R \sqsubseteq R \mid R \text{ is a role}\}$.

The semantics of $\mathcal{SHOI}$ is defined by an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ given by a pair of a non-empty set $\Delta^\mathcal{I}$, referred to as the *domain* of interpretation, and an interpretation function $\cdot^\mathcal{I}$. The function $\cdot^\mathcal{I}$ maps individuals to elements of the domain, concept names to subsets of $\Delta^\mathcal{I}$ and role names to binary relations over $\Delta^\mathcal{I}$. Regarding roles declared as being transitive, $\cdot^\mathcal{I}$ must satisfy that $r^\mathcal{I}$ is a transitive relation whenever $\mathsf{Trans}(r)$ is true. The function $\cdot^\mathcal{I}$ extends to all concept and role expressions by induction on lengths of expressions as follows:

$$a^\mathcal{I} \overset{\mathsf{def}}{=} \{a^\mathcal{I}\}, \quad (\neg C)^\mathcal{I} \overset{\mathsf{def}}{=} \Delta^\mathcal{I} \setminus C^\mathcal{I}, \quad (C \sqcup D)^\mathcal{I} \overset{\mathsf{def}}{=} C^\mathcal{I} \cup D^\mathcal{I},$$

$$(\exists R.C)^\mathcal{I} \overset{\mathsf{def}}{=} \{x \mid \exists y \in C^\mathcal{I}\ (x, y) \in R^\mathcal{I}\}, \qquad (R^-)^\mathcal{I} \overset{\mathsf{def}}{=} \{(x, y) \mid (y, x) \in R^\mathcal{I}\}.$$

According to the semantics, the inverse of a role is transitive iff the role is transitive. Following this, we extend the predicate Trans to all role expressions so that $\mathsf{Trans}(r^-)$ is true iff $\mathsf{Trans}(r)$ is true.

Let E denote any concept expression, any concept inclusion, any role inclusion, any concept assertion or any role assertion. We indicate by $\mathcal{I} \models E$ that E is *valid* in the model $\mathcal{I}$. We define:

$$\begin{array}{llll}
\mathcal{I} \models C & \overset{\text{def}}{\iff} C^{\mathcal{I}} = \Delta^{\mathcal{I}} & \mathcal{I} \models a : C & \overset{\text{def}}{\iff} a^{\mathcal{I}} \in C^{\mathcal{I}} \\
\mathcal{I} \models R \sqsubseteq S & \overset{\text{def}}{\iff} R^{\mathcal{I}} \subseteq S^{\mathcal{I}} & \mathcal{I} \models (a,b) : R & \overset{\text{def}}{\iff} (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}} \\
\mathcal{I} \models C \sqsubseteq D & \overset{\text{def}}{\iff} C^{\mathcal{I}} \subseteq D^{\mathcal{I}} & &
\end{array}$$

Because $\mathcal{SHOI}$ supports singleton concepts, every ABox statement $a : C$ can be encoded by the TBox statement $\{a\} \sqsubseteq C$. Also, every role assertion $(a,b) : R$ can be encoded as the TBox statement $\{a\} \sqsubseteq \exists R.\{b\}$. Thus, without loss of generality, we assume that a knowledge base is a pair $(\mathcal{T}, \mathcal{R})$ which consists of a normalised TBox $\mathcal{T}$ and an RBox $\mathcal{R}$. It worth noting that the TBox can be internalised as well [15] but for performance reasons we present a tableau calculus that handles TBox statements directly.

A concept C is *satisfiable* in a model $\mathcal{I}$ iff $C^{\mathcal{I}} \neq \emptyset$. A concept is *satisfiable in* $\mathcal{I}$ *with respect to a knowledge base* if it is satisfiable in $\mathcal{I}$ whenever every statement of the knowledge base is valid in $\mathcal{I}$. That is, C is satisfiable with respect to $(\mathcal{T}, \mathcal{R})$ in $\mathcal{I}$ iff $C^{\mathcal{I}} \neq \emptyset$ provided that $\mathcal{I} \models E$ for every $E \in \mathcal{T} \cup \mathcal{R}$.

3 An Abstract Tableau Calculus for $\mathcal{SHOI}$

In this section we present a labelled semantic ground tableau calculus for $\mathcal{SHOI}$.

The language of the tableau calculus is an extension of the language of $\mathcal{SHOI}$ with equality formulae and individual terms used as labels. We add a function symbol f which takes a triple (s, R, C) consisting of an individual term s, a role expression R and a concept expression C as its arguments and define the set of *(individual) terms* s inductively by the following grammar rule, where a denotes any individual, C any concept and R any role.

$$s \overset{\text{def}}{=} a \mid f(s, R, C)$$

Terms which are not ABox individuals can be viewed as being Skolem terms.

Formulae in the tableau language are defined by the following grammar rule, where s and t are individual terms, C is a concept and R is a role.

$$E \overset{\text{def}}{=} s : C \mid (s,t) : R \mid s \approx t$$

We extend the interpretation of $\mathcal{SHOI}$ expressions to the formulae of the tableau language. For every $\mathcal{SHOI}$ interpretation $\mathcal{I}$, let the interpretation $f^{\mathcal{I}}$ in $\mathcal{I}$ of the function f be an arbitrary function mapping triples (x, ρ, χ) with $x \in \Delta^{\mathcal{I}}$, $\rho \subseteq (\Delta^{\mathcal{I}})^2$, $\chi \subseteq \Delta^{\mathcal{I}}$ to elements of $\Delta^{\mathcal{I}}$. We let

$$\begin{array}{rclrcl}
(f(a,R,C))^{\mathcal{I}} & \overset{\text{def}}{=} & f^{\mathcal{I}}(a^{\mathcal{I}}, R^{\mathcal{I}}, C^{\mathcal{I}}), & \mathcal{I} \models (s : C)^{\mathcal{I}} & \overset{\text{def}}{\iff} & s^{\mathcal{I}} \in C^{\mathcal{I}}, \\
\mathcal{I} \models s \approx t & \overset{\text{def}}{\iff} & s^{\mathcal{I}} = t^{\mathcal{I}}, & \mathcal{I} \models (s,t) : R & \overset{\text{def}}{\iff} & (s^{\mathcal{I}}, t^{\mathcal{I}}) \in R^{\mathcal{I}}.
\end{array}$$

The interpretations of the formulae $s \approx t$, $s : \{t\}$ and $t : \{s\}$ coincide. In accordance with their interpretation we refer to these formulae as *equalities*. We also refer to the formulae of the form $s : \neg\{t\}$ as *inequalities*.

Let *Tab* denote a tableau calculus comprising of a set of inference rules. A *derivation* or *tableau* for *Tab* is a finitely branching, ordered tree whose nodes are annotated by sets of tableau formulae. Assuming that C is the input concept expression to be tested for satisfiability with respect a knowledge base $(\mathcal{T}, \mathcal{R})$ the root node of the tableau is the set $\{a : C\}$, where a denotes a fresh individual. Successor nodes are constructed in accordance with a set of inference rules in the calculus. The inference rules have the general form

$$\frac{X_0}{X_1 \mid \ldots \mid X_n} \text{ (side-condition)},$$

where X_0 is the set of premises and the X_i are the sets of conclusions. If $n = 0$, the rule is called *closure rule* and written $X_0/\bot$.

If σ is a substitution that acts on tableau formulae and $X = \{E_1, \ldots, E_k\}$ is a set of tableau formulae then $X\sigma$ denotes the set $\{E_1\sigma, \ldots, E_k\sigma\}$. A rule is *applicable* to a tableau if there is a leaf node annotated with a set N and there is a substitution σ such that $X_0\sigma \subseteq N$, where X_0 is the set of premises of the rule, and the side-condition of the rule is true for N. σ is called the *matching substitution* of the rule application. We assume in a rule individual symbols, concept symbols and role symbols represent variables that are matched with individual terms, concept expressions and role expressions respectively. We also say the rule is applicable to the formulae $X_0\sigma$ in (the leaf node of) the branch.

If a rule of the calculus is applicable to a leaf node of the tableau with a matching substitution σ, then the tableau is extended by attaching to the leaf node n child nodes annotated with $N \cup X_i\sigma$ for $i = 1, \ldots n$, respectively. In order to avoid redundancies we stipulate that a rule application to a leaf node annotated with N is *redundant* if there is a conclusion set X_i for some $i = 1, \ldots n$ of the rule such that $X_i\sigma \subseteq N$, where σ is the matching substitution. This ensures rules are not applied more than once to the same sets of formulae.

A *branch* in the tableau is a maximal path from the root of the tableau to a leaf node. If a closure rule has been applied in a branch then the branch is said to be *closed*. If a branch is not closed, it is called *open*. A tableau is *closed* if all its branches are closed. A branch is *fully expanded* if no more rules are applicable to its leaf node modulo redundancy. We call a tableau *fully expanded* iff all its branches are fully expanded. We denote by *Tab*$(\mathcal{T}, \mathcal{R}, C)$ a fully expanded tableau constructed in the calculus *Tab* for the input concept C and the knowledge base $(\mathcal{T}, \mathcal{R})$.

We need equality reasoning for individual terms to achieve termination for the calculus. Equality reasoning can be provided in various ways. One is to supply special tableau rules for reasoning modulo equalities within the branch in a similar way as is done in [3,14,13]. Another is to use ordered term rewriting. Ordered rewriting is more efficient for handling equal individuals because it allows to reduce the number of tableau formulae in the current branch. Since

$$(\bot)\text{: } \frac{s:\neg C,\ s:C}{\bot}$$

$$(\sqcup)\text{: } \frac{s:C\sqcup D}{s:C \mid s:D}$$

$$(\exists)\text{: } \frac{s:\exists R.C}{f(s,R,C):C,\ (s,f(s,R,C)):R}$$

$$(\neg\exists)\text{: } \frac{s:\neg\exists S.C,\ (s,t):R}{t:\neg C}\ (R\sqsubseteq S\in\mathcal{R}^*)$$

$$(\neg\exists^-)\text{: } \frac{s:\neg\exists S^-.C,\ (t,s):R}{t:\neg C}\ (R\sqsubseteq S\in\mathcal{R}^*)$$

$$(^+)\text{: } \frac{s:\neg\exists S.C,\ (s,t):R}{t:\neg\exists R.C}\ (R\sqsubseteq S\in\mathcal{R}^*,\ \mathsf{Trans}(R)\in\mathcal{R})$$

$$(^{-+})\text{: } \frac{s:\neg\exists S^-.C,\ (t,s):R}{t:\neg\exists R^-.C}\ (R\sqsubseteq S\in\mathcal{R}^*,\ \mathsf{Trans}(R)\in\mathcal{R})$$

$$(\mathsf{RBox})\text{: } \frac{(s,t):R}{(s,t):S}\ (R\sqsubseteq S\in\mathcal{R}^+)$$

$$(\neg\neg)\text{: } \frac{s:\neg\neg C}{s:C}$$

$$(\neg\sqcup)\text{: } \frac{s:\neg(C\sqcup D)}{s:\neg C,\ s:\neg D}$$

$$(^-)\text{: } \frac{(s,t):R^-}{(t,s):R}$$

$$(\mathsf{id})\text{: } \frac{s:C}{s:\{s\}}$$

$$(\mathsf{id}_2)\text{: } \frac{s:\neg\{t\}}{t:\{t\}}$$

$$(\mathsf{cng})\text{: } \frac{(s,t):R}{s:\{s\},\ t:\{t\}}$$

$$(\mathsf{TBox})\text{: } \frac{s:\{s\}}{s:C}\ (C\in\mathcal{T})$$

$$(\approx)\text{: } \frac{s:\{t\}}{s\approx t}\ (s\neq t)$$

Figure 1. The tableau calculus $Tab_{\mathcal{SHOI}}$

all individual terms in any tableau derivation are ground we are dealing with a special case of rewriting, namely, ground rewriting.

In this paper, a *rewrite system* R is a binary relation on the set of all individual terms and consists of rewrite rules which are pairs of individual terms. In order to handle equalities, we orient each equality formula appearing in the current branch of a tableau derivation according to a special ordering $\succ$ which is a strict partial order on individual terms. We denote by $s \rightarrow t$ a rewrite rule (s,t) in which $s \succ t$. Thus, if an equality formula $s \approx t$ appears in a node of a branch then either $s \rightarrow t$ or $t \rightarrow s$ is added as a rewrite rule to the rewrite system of the branch. A term which cannot be rewritten (with respect to a rewrite system) is said to be in *normal form*. A normal form of a term s is denoted by $\mathsf{nf}(s)$. A rewrite system is *terminating* if there is a normal form for each term.

Our tableau calculus $Tab_{\mathcal{SHOI}}$ for the description logic $\mathcal{SHOI}$ is given in Figure 1. The $(\bot)$ rule is the closure rule. The $(\neg\neg)$ rule removes occurrences of double negation on concepts. The $(\sqcup)$ and $(\neg\sqcup)$ rules are standard rules for handling concept disjunctions. Given a tableau formula $s : \exists R.C$, the $(\exists)$ rule introduces an individual term $f(s,R,C)$ as an R-successor of s (instead of introducing a fresh individual as might be done in other presentations). The $(\neg\exists)$ rule is equivalent to the standard rule for universally restricted concept expressions. The $(\neg\exists^-)$ rule allows the backward propagation of concept expressions along inverted links. The $(^-)$ rule inverts a given link.

The $(^+)$ rule propagates negated existential concept restriction along a transitive link while the $(^{-+})$ rule does the same for inverse occurrences of transitive roles. Equalities of the form $s : \{s\}$ are tautologies, used in our calculus as domain predicates for keeping track of the terms that have been introduced to a branch. This is achieved with the three rules (id), (id$_2$) and (cng). The (TBox) rule concatenates every concept of the normalised TBox with every label occurring on the branch. The (RBox) propagates a link of a role into its superrole according to the closure $\mathcal{R}^+$ of the given RBox $\mathcal{R}$.

The ($\approx$) rule is a special rule adding, what we call, a *rewrite trigger* $s \approx t$ to the branch. Let $\succ$ be any reduction ordering on the set of individuals in the branch. The addition of any tableau formula $s \approx t$ to a set N of formulae which annotates a leaf tableau node immediately triggers the following rewrite process. Suppose that $s \succ t$ (the case $t \succ s$ is symmetrical). Then, $s \rightarrow t$ is added to a rewrite system R associated with the current tableau branch. The tableau is extended by attaching one child node to the current leaf node. The child node is annotated by the set N' obtained by rewriting all the tableau formulae in N with respect to the rewrite system R. In particular, this means that, in N' every term s is replaced by a term u such that $s \overset{*}{\rightarrow} u$ with respect to R.

4 Soundness, Completeness and Termination

It is not difficult to see that each rule of $\mathit{Tab}_{\mathcal{SHOI}}$ is sound, i.e., preserves satisfiability of concept assertions. Consequently:

Theorem 1 (Soundness). *The tableau calculus $\mathit{Tab}_{\mathcal{SHOI}}$ is sound for the description logic $\mathcal{SHOI}$. That is, if a concept C is satisfiable with respect to the knowledge base $(\mathcal{T}, \mathcal{R})$ then any fully expanded $\mathit{Tab}_{\mathcal{SHOI}}$-tableau for $(\mathcal{T}, \mathcal{R}, C)$ has an open branch.*

A tableau calculus *Tab* is *complete* iff for every knowledge base $(\mathcal{T}, \mathcal{R})$ and every concept C if C is unsatisfiable with respect to $(\mathcal{T}, \mathcal{R})$ then there is a closed tableau $\mathit{Tab}(\mathcal{T}, \mathcal{R}, C)$. In order to prove completeness of $\mathit{Tab}_{\mathcal{SHOI}}$, we prove its constructive completeness which implies completeness. A tableau calculus *Tab* is *constructively complete* if for every open branch in any fully expanded tableau $\mathit{Tab}(\mathcal{T}, \mathcal{R}, C)$ there is a model which validates the knowledge base $(\mathcal{T}, \mathcal{R})$ and satisfies C.

Theorem 2 (Completeness). *$\mathit{Tab}_{\mathcal{SHOI}}$ is a (constructively) complete tableau calculus for the description logic $\mathcal{SHOI}$.*

Next, we establish termination. A tableau calculus *Tab* is *(weakly) terminating* if any tableau $\mathit{Tab}(\mathcal{T}, \mathcal{R}, C)$ has a finite open branch provided that C is satisfiable concept with respect to the knowledge base $(\mathcal{T}, \mathcal{R})$.

Although $\mathit{Tab}_{\mathcal{SHOI}}$ is a sound and complete tableau calculus for the description logic $\mathcal{SHOI}$, it is not terminating. In order to achieve termination, a form of blocking or loop-checking is necessary. One possibility is to add the *unrestricted blocking mechanism* described in [14]. As is shown in [12], this will ensure termination of an arbitrary tableau calculus under certain conditions, one of which being condition (c2) discussed below.

In this paper, we take a slightly different route and introduce a modified version of the unrestricted blocking mechanism. It is given by the (ub-rw) rule and ordered rewriting.

$$\text{(ub-rw)}: \frac{s : \{s\},\ t : \{t\}}{s \approx t \mid s : \neg\{t\}} \ (s \neq t)$$

Here, $s \approx t$ is a rewrite trigger as introduced in Section 3. Premises of this rule are instantiated with any two distinct terms s and t used as labels in a set of tableau formulae N annotating the current leaf node. As a result of a rule application two successor nodes are created. If $s \succ t$ (respectively $t \succ s$) then in the left node a rewrite rule $s \rightarrow t$ (respectively $t \rightarrow s$) is added to the rewrite system $\mathcal{R}$. The left node is annotated with a copy of N, which is rewritten with respect to the newly obtained rewrite system $\mathcal{R}$. The right node is annotated with a copy of N extended with the additional formula $s : \neg\{t\}$. This formula indicates the case that s and t are not equal.

The calculus consisting of all the rules of $Tab_{\mathcal{SHOI}}$ and the rule (ub-rw) is denoted by $Tab_{\mathcal{SHOI}}$(ub-rw). Clearly, the (ub-rw) rule is sound. Therefore:

Theorem 3. *The calculus* $Tab_{\mathcal{SHOI}}$(ub-rw) *is a sound and (constructively) complete for the description logic* $\mathcal{SHOI}$.

In order to ensure termination for a procedure based on $Tab_{\mathcal{SHOI}}$(ub-rw) the rule application strategy must satisfy the following condition.

(c2) In every open branch there is some node from which point onward before any application of the ($\exists$) rule all possible applications of the (ub-rw) rule have been performed.

(The unrestricted blocking mechanism in [14] also needs to satisfy a second condition, which is already satisfied in our modified setting.)

Provided that condition (c2) holds, a sufficient and necessary condition for termination of the tableau procedures based on $Tab_{\mathcal{SHOI}}$(ub-rw) is that $\mathcal{SHOI}$ has the finite model property with respect to its standard semantics. This can be shown in a similar way as in [13]. A description logic has the *finite model property* if for an arbitrary concept C and arbitrary knowledge base it holds that if C is satisfiable with respect to the knowledge base in a model for the logic then C is satisfiable with respect to the knowledge base in a *finite* model of the logic.

The finite model property for $\mathcal{SHOI}$ can be shown by a filtration argument.

Theorem 4 (Finite model property of $\mathcal{SHOI}$). *The description logic* $\mathcal{SHOI}$ *has the finite model property.*

Therefore, using the results of [13] we obtain the following theorem.

Theorem 5 (Termination). *Any implementation, fair in the sense of [13], of the tableau calculus* $Tab_{\mathcal{SHOI}}$(ub-rw) *and satisfies condition (c2) is a decision procedure for* $\mathcal{SHOI}$ *and its sublogics.*

5 Sound Restricted Blocking

The (ub-rw) rule creates potentially many branching points in a derivation, especially if the number individuals and $\exists$-expressions in the knowledge base is high. A way to reduce the number of applications of the (ub-rw) rule and thus reduce the search space is to apply the blocking rule less often. This can be achieved by adding side-conditions and/or premises to the rule. Ideal would be side-conditions and additional premises that maximise the chance of constructing a finite model without the need for backtracking.

In the remaining section, we give some examples of restricted versions of the (ub-rw) rule. They all preserve soundness and completeness. We have:

Theorem 6 (Soundness and completeness). *The* (ub-rw) *rule constrained by additional premises or side-conditions is sound. Thus,* Tab$_{\mathcal{SHOI}}$ *extended with such a modified rule is sound and complete for* $\mathcal{SHOI}$.

Most existing description logic tableau algorithms aim to construct models given by relational tree structures where the nodes are individuals (or individual terms, if Skolem terms would have been used) and are annotated with label sets of concept expressions. A *label set* of a term s is the set $L(s) \overset{\mathsf{def}}{=} \{C \mid s : C \in N\}$. These label sets are then used in the tests of standard blocking mechanism such as subset ancestor blocking and dynamic anywhere equality blocking.

An *emulation of subset ancestor blocking* can be realised through the selective application of the (ub-rw) rule, realised by adding a side-condition:

$$\text{(ub}_{\subseteq}\text{-rw)}: \frac{s : \{s\},\ t : \{t\}}{s \approx t \mid s : \neg\{t\}} \ (s \neq t,\ s \text{ is an ancestor of } t \text{ and } L(t) \subseteq L(s))$$

In this rule the application of the (ub-rw) rule is restricted to a term s and its successor term t, where the label set of s is a superset of the label set of t. In our setting, as the calculus creates Skolem terms in the ($\exists$) rule, a term s is an ancestor of a term t, if s is a subterm of t.

Standard ancestor subset blocking is used in tableau algorithms for description logics $\mathcal{ALC}$, $\mathcal{S}$ and $\mathcal{SH}$ [7]. In ancestor subset blocking, a term t is blocked by its ancestor s if $L(t) \subseteq L(s)$. No rule is applicable to the blocked individuals. As standard ancestor blocking is not a branching rule it is important to perform the expansions in a stratified way and perform the subset test at an appropriate moment in order to preserve soundness. But even if the expansions are performed in the required way standard ancestor blocking is not generally sound unlike blocking based on the (ub$_{\subseteq}$-rw) rule.

Application of the (ub-rw) rule can be limited by ignoring the pairs of terms where the application of the rule is not critical for termination. E.g., it is possible to ignore pairs where both terms appear before some fixed node of a tableau derivation. We believe, as there are a finite number of individuals before a fixed tableau node, excluding them does not endanger termination. In particular, the pairs where both terms are ABox individuals can be ignored as in [8]. If the unique name assumption is assumed for the given ABox individuals, identifying these individuals by blocking would be incorrect. Using the following rule instead of using the (ub-rw) rule can have a significant impact on the performance, especially when reasoning over knowledge bases with a large number of individuals.

$$\text{(ub}_{\text{No ABox}}\text{-rw)}: \frac{s : \{s\},\ t : \{t\}}{s \approx t \mid s : \neg\{t\}} \ (s \neq t, \text{ not both } s \text{ and } t \text{ are ABox individuals})$$

We can define a variation of the (ub-rw) rule restricted to the terms that are known to be the ones that may cause infinite derivations. For $Tab_{\mathcal{SHOI}}$, infinite derivations can be caused only by infinite applications of the ($\exists$) rule. This means we may focus blocking on the terms to which the ($\exists$) rule may eventually be applicable, i.e., the terms which have an $\exists$-expression in their label sets. We may formulate the (ub-rw) rule as follows to reflect this restriction:

$$\text{(ub}_{\exists}\text{-rw):}\ \frac{s : \exists R.C,\ t : \exists S.D}{s \approx t \mid s : \neg\{t\}}\ (s \neq t)$$

Here, $\exists R.C$, $\exists S.D$ are two $\exists$-expression that can be matched with any $\exists$-expression. This rule is applicable to any pair of terms s and t which both have a $\exists$-expression in their label sets.

The three variations of the (ub-rw) rule just presented are all sound, thus preserving soundness (and completeness) of the calculus is not an issue. An issue is to show under which conditions and for which logics termination can be ensured. Because of the side-conditions or additional premises these variations of the (ub-rw) rule are no longer applied to every possible pair of terms. Thus, condition (c2) does not hold. We believe however it can be proved that search strategies can be adopted where blocking applies to sufficiently many pairs so that the procedure terminates.

Next we illustrate how the (δ^*) rule [6] can be simulated using a restriction of the (ub-rw) rule. The (δ^*) rule systematically reuses terms in order to find finite models. For description logics the (δ^*) rule is defined as follows:

$$(\delta^*)\text{:}\ \frac{s : \exists R.C}{(s, t_1) : R,\ t_1 : C \mid \cdots \mid (s, t_n) : R,\ t_n : C \mid (s, f(s, R, C)) : R,\ f(s, R, C) : C}$$

Here, $t_1, \ldots, t_n$ are all existing terms, covering all given ABox individuals and all introduced Skolem terms on the current branch. The (δ^*) rule is actually a modified version of the ($\exists$) rule. Instead of creating a new term to satisfy an $\exists$-expression, this rule tries to satisfy the $\exists$-expression by reusing existing terms. If all the attempts to satisfy the $\exists$-expression with existing terms end in contradictions, then a new term $f(s, R, C)$ is introduced.

In our abstract calculus we can simulate the (δ^*) rule with the ($\exists$) rule and modifying the (ub-rw) rule to:

$$\text{(ub}_{\delta^*}\text{-rw):}\ \frac{s : \{s\},\ t : \{t\}}{s \approx t \mid s : \neg\{t\}}\ (s \neq t,\ t \text{ is a Skolem term})$$

We should use a rule application strategy where after each application of the ($\exists$) rule, the (ub$_{\delta^*}$-rw) rule is applied to the newly added Skolem term and every existing term. In contrast to the previous blocking variations, the (ub$_{\delta^*}$-rw) rule satisfies condition (c2), since all possible term comparisons are performed before any application of the ($\exists$) rule. Hence termination is ensured.

Theorem 7 (Termination). *Any implementation, fair in the sense of [13], of the tableau calculus* $\mathrm{Tab}_{\mathcal{SHOI}}$ *extended with the* (ub$_{\delta^*}$-rw) *rule and using the described strategy is a decision procedure for* $\mathcal{SHOI}$ *and its sublogics.*

6 Concluding Remarks

The contribution of this paper is an abstract labelled tableau calculus for the description logic $\mathcal{SHOI}$ using ordered rewriting and generic forms of blocking defined as variations of the unrestricted blocking mechanism. The tableau calculus is designed to be as general as possible in order to gain greater insight into minimal requirements for soundness, completeness and termination and conduct the proofs without any considerations for search strategies, heuristics and other implementation issues. The discussion in [13] of how to obtain deterministic tableau procedures for implementation based on the notion of fairness as defined in that paper carries over to the calculi presented here. We hope this ongoing work will lead to even greater insight of the theory and techniques of different tableau approaches for description logics and their implementation.

References

1. Alenda, R., Olivetti, N., Schwind, C., Tishkovsky, D.: Tableau calculi for CSL over min-spaces. In: Proc. CSL'10. LNCS, vol. 6247, pp. 52–66. Springer (2010)
2. Baader, F., Sattler, U.: An overview of tableau algorithms for description logics. Studia Logica 69(1), 5–40 (2001)
3. Bolander, T., Blackburn, P.: Termination for hybrid tableaus. J. Logic Comput. 17(3), 517–554 (2007)
4. Cialdea Mayer, M., Cerrito, S.: Nominal substitution at work with the global and converse modalities. In: Proc. AiML-8. pp. 57–74. College Publ. (2010)
5. Fitting, M.: Proof methods for modal and intuitionistic logics. Kluwer (1983)
6. Hintikka, J.: Model minimization: An alternative to circumscription. J. Automat. Reason. 4(1), 1–13 (1988)
7. Horrocks, I.: Using an expressive description logic: FaCT or fiction? In: Proc. KR-98. pp. 636–647. Morgan Kaufmann (1998)
8. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: Proc. KR 2006. pp. 57–67. AAAI Press (2006)
9. Horrocks, I., Sattler, U.: A description logic with transitive and inverse roles and role hierarchies. J. Logic Comput. 9(3), 385–410 (1999)
10. Horrocks, I., Sattler, U.: A tableau decision procedure for SHOIQ. J. Automat. Reason. 39(3), 249–276 (2007)
11. Schmidt, R.A., Tishkovsky, D.: A general tableau method for deciding description logics, modal logics and related first-order fragments. In: Proc. IJCAR'08. LNCS, vol. 5195, pp. 194–209. Springer (2008)
12. Schmidt, R.A., Tishkovsky, D.: Automated synthesis of tableau calculi. Logical Methods in Comput. Sci. 7(2), 1–32 (2011)
13. Schmidt, R.A., Tishkovsky, D.: Using tableau to decide description logics with full role negation and identity (2011), manuscript, `http://www.mettel-prover.org/papers/ALBOid.pdf`
14. Schmidt, R.A., Tishkovsky, D.: Using tableau to decide expressive description logics with role negation. In: Proc. ISWC+ASWC'07. pp. 438–451. Springer (2007)
15. Tobies, S.: The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. J. Artificial Intelligence Res. 12, 199–217 (2000)