EXPTIME Tableaux with Global Caching for Description Logics with Transitive Roles, Inverse Roles and Role Hierarchies

Rajeev Goré[1,⋆] and Linh Anh Nguyen[2]

[1] The Australian National University and NICTA
Canberra ACT 0200, Australia
Rajeev.Gore@anu.edu.au
[2] Institute of Informatics, University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland
nguyen@mimuw.edu.pl

Abstract. The description logic $\mathcal{SHI}$ extends the basic description logic $\mathcal{ALC}$ with transitive roles, role hierarchies and inverse roles. The known tableau-based decision procedure [9] for $\mathcal{SHI}$ exhibit (at least) NEXPTIME behaviour even though $\mathcal{SHI}$ is known to be EXPTIME-complete. The automata-based algorithms for $\mathcal{SHI}$ often yield optimal worst-case complexity results, but do not behave well in practice since good optimisations for them have yet to be found. We extend our method for global caching in $\mathcal{ALC}$ to $\mathcal{SHI}$ by adding analytic cut rules, thereby giving the first EXPTIME tableau-based decision procedure for $\mathcal{SHI}$, and showing one way to incorporate global caching and inverse roles.

1 Introduction and Motivation

Description logics (DLs) are notational variants of multi-modal logics which have proved to be important in representing and reasoning about knowledge. We assume the reader is familiar with the notions of transitive roles, inverse roles, role hierarchies and TBoxes (global assumptions) in DLs [1].

The decision problem for most of these logics (with global assumptions) is EXPTIME-hard. The known decision procedures for these logics are tableau-based, resolution-based or automata-based. In practice, the automata-based methods do not behave as well as the other methods since good optimisations for them have not been found to date. For very expressive description logics, the most successful practical methods are all tableau-based.

Global assumptions or transitive roles can cause the traditional tableau methods for these logics to contain infinite branches because certain nodes repeat *ad infinitum*. The usual solution is to remember certain nodes created along the current branch using "histories" and to "block" all rules that would re-create an

⋆ National ICT Australia is funded by the Australian Government's Dept of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence program.

N. Olivetti (Ed.): TABLEAUX 2007, LNAI 4548, pp. 133–148, 2007.

exact copy of any remembered nodes [1,7]. For inverse roles, blocked nodes can become unblocked, and reblocked later, requiring "dynamic blocking" [9].

The basic description logic $\mathcal{ALC}$ extended with transitive roles, inverse roles, and role hierarchies is called $\mathcal{SHI}$. Horrocks and Sattler [9] gave a NEXPTIME tableau decision procedure for $\mathcal{SHI}$. In [1], Baader and Sattler wrote: *"The point in designing these algorithms* [for $\mathcal{SHI}$] *was not to prove worst-case complexity results, but ⋯ to obtain 'practical' algorithms ⋯ that are easy to implement and optimise, and which behave well on realistic knowledge bases. Nevertheless, the fact that 'natural' tableau algorithms for such EXPTIME-complete logics are usually NEXPTIME-algorithms is an unpleasant phenomenon. ⋯ Attempts to design EXPTIME-tableaux for such logics (De Giacomo et al., 1996; De Giacomo and Massacci, 1996; Donini and Massacci, 1999) usually lead to rather complicated (and thus not easy to implement) algorithms, which (to the best of our knowledge) have not been implemented yet."* [1, page 26].

More precisely, the mentioned work of De Giacomo and Massacci [2] gives only a NEXPTIME tableau decision procedure for propositional dynamic logic with converse (CPDL) and an informally described transformation of NEXPTIME tableaux into an EXPTIME decision procedure for CPDL using the "look behind analytic cut". Using analytic cuts to eliminate nondeterminism and reduce the complexity from NEXPTIME to EXPTIME is a good idea. But it is not clear how to eliminate the nondeterminism (of the NEXPTIME decision procedure for CPDL [2]) using "look behind analytic cut". Specifically, suppose we apply an or-rule to a node w and obtain two successor nodes u and v, and after that we follow the u-branch. If we later "look behind" and add a formula φ to w, then φ affects node v. But how to eliminate the or-branching at w ?

Thus, to the best of our knowledge, no "natural and easy to implement" EXPTIME tableau decision procedures have been given or implemented for $\mathcal{SHI}$. Here, we give such a procedure using analytic cuts to "guess the future" so we *never look behind*, in unison with our method of sound global caching for $\mathcal{ALC}$ [6].

Caching is a crucial optimisation technique for obtaining efficient tableau-based decision procedures for description logics [8,3]. Naive use of caching can be unsound, so complex and difficult to implement methods have been devised to regain EXPTIME decision procedures for the basic logic $\mathcal{ALC}$ [4]. But the decision procedure given by Donini and Massacci [4] for $\mathcal{ALC}$ permanently caches *"all and only unsatisfiable sets of concepts"*, and temporarily caches visited nodes on the current branch, even though this means that *"many potentially satisfiable sets of concepts are discarded when passing from a branch to another branch"*. Even more complicated methods have been suggested for handling inverse roles and transitive roles, but these require severe restrictions to retain soundness [3]. Our global caching method is based on building a simple and-or graph using a sound and complete traditional tableau calculus for $\mathcal{ALC}/\mathcal{SHI}$.

Our method for transforming our tableau calculus for $\mathcal{SHI}$ into an EXPTIME decision procedure for $\mathcal{SHI}$ is almost the same as that for $\mathcal{ALC}$ in our previous work [6]. The minor differences are some alterations for $\mathcal{SHI}$ and that some of the tableau rules for $\mathcal{SHI}$ carry their principal concepts into their denominators

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \qquad \bot^{\mathcal{I}} = \emptyset \qquad (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \qquad (r^-)^{\mathcal{I}} = (r^{\mathcal{I}})^{-1}$$
$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \qquad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$$
$$(C \sqsubseteq D)^{\mathcal{I}} = (\neg C \sqcup D)^{\mathcal{I}} \qquad (C \doteq D)^{\mathcal{I}} = ((C \sqsubseteq D) \sqcap (D \sqsubseteq C))^{\mathcal{I}}$$
$$(\forall R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b.(a,b) \in R^{\mathcal{I}} \text{ implies } b \in C^{\mathcal{I}}\}$$
$$(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}} \text{ and } \ b \in C^{\mathcal{I}}\}$$

Fig. 1. Interpretation of Concepts and Roles

(which we need for combining the proofs of completeness of the calculus with correctness of global caching). The overlap is needed to make this paper self-contained. The major differences of this paper w.r.t. [6] lie in the use of the cut rules, and the non-trivial complications necessary to ensure that the resulting calculus and the algorithm resulting from it are complete.

In Section 2, we recall the notation and semantics of $\mathcal{SHI}$. In Section 3, we present our tableau calculus for $\mathcal{SHI}$ and prove its soundness, and in Section 4, we prove its completeness. In Section 5, we present a simple EXPTIME decision procedure for $\mathcal{SHI}$ that is based on the calculus and uses global caching and propagation. In Section 6, we outline further work and conclusions.

2 Notation and Semantics of $\mathcal{SHI}$

Given a finite set RNames of role names, let $\text{RNames}^- = \{r^- \mid r \in \text{RNames}\}$ be the set of *inverse roles*. A $\mathcal{SHI}$ *role* is any member of $\text{RNames} \cup \text{RNames}^-$. We use uppercase letters like R and S for $\mathcal{SHI}$ roles. For any $\mathcal{SHI}$ role R the inverse role $R^- = r^-$ if $R = r$ and $R^- = r$ if $R = r^-$.

We use A for an atomic concept and use C and D for arbitrary concepts. Concepts in $\mathcal{SHI}$ are formed using the following BNF grammar:

$$C, D ::= \top \mid \bot \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid C \sqsubseteq D \mid C \doteq D \mid \forall R.C \mid \exists R.C$$

An *interpretation* $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ consists of a non-empty set $\Delta^{\mathcal{I}}$, the *domain* of $\mathcal{I}$, and a function $\cdot^{\mathcal{I}}$, the *interpretation function* of $\mathcal{I}$, that maps every atomic concept to a subset of $\Delta^{\mathcal{I}}$ and every role name to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The interpretation function is extended to interpret all $\mathcal{SHI}$ concepts and roles as shown in Figure 1. An interpretation $\mathcal{I}$ *satisfies* a concept C if $C^{\mathcal{I}} \neq \emptyset$, and *validates* C if $C^{\mathcal{I}} = \Delta^{\mathcal{I}}$. Clearly, $\mathcal{I}$ *validates* C iff it does not *satisfy* $\neg C$.

A TBox (of global axioms/assumptions) $\mathcal{T}$ is a finite set of concepts. (Traditionally, $C \sqsubseteq D$ and $C \doteq D$ are not treated as concept constructors, and a TBox is defined to be a finite set of terminological axioms of the form $C \sqsubseteq D$ or $C \doteq D$, where C and D are concepts, but the two definitions are equivalent.) An interpretation $\mathcal{I}$ is a *model* of $\mathcal{T}$ if $\mathcal{I}$ validates all concepts in $\mathcal{T}$.

A $\mathcal{SHI}$ *RBox* $\mathcal{R}$ is a finite set of role inclusion axioms $R \sqsubseteq S$ and transitivity axioms $R \circ R \sqsubseteq R$ for $\mathcal{SHI}$ roles R and S satisfying:

RBox 1: $R \sqsubseteq R \in \mathcal{R}$ for every $\mathcal{SHI}$-role R,
RBox 2: $R \sqsubseteq S \in \mathcal{R}$ implies $R^- \sqsubseteq S^- \in \mathcal{R}$,
RBox 3: $R \circ R \sqsubseteq R \in \mathcal{R}$ implies $R^- \circ R^- \sqsubseteq R^- \in \mathcal{R}$,
RBox 4: $R \sqsubseteq R' \in \mathcal{R}$ and $R' \sqsubseteq R'' \in \mathcal{R}$ imply $R \sqsubseteq R'' \in \mathcal{R}$, and
RBox 5: $R \sqsubseteq S \in \mathcal{R}$ and $S \sqsubseteq R \in \mathcal{R}$ imply $R = S$.

An interpretation $\mathcal{I}$ is a *model* of $\mathcal{R}$ if $\mathcal{I}$ validates all axioms in $\mathcal{T}$: that is, $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ if $R \sqsubseteq S \in \mathcal{R}$, and $R^{\mathcal{I}} \circ R^{\mathcal{I}} \subseteq R^{\mathcal{I}}$ if $R \circ R \sqsubseteq R \in \mathcal{R}$.

We use X, Y for finite sets of concepts. We say that $\mathcal{I}$ *satisfies* X if there exists $d \in \Delta^{\mathcal{I}}$ such that $d \in C^{\mathcal{I}}$ for all $C \in X$. Note: by our definition, satisfaction is defined "locally", and $\mathcal{I}$ satisfies X does not mean that $\mathcal{I}$ is a model of X.

We say that $(\mathcal{R}, \mathcal{T})$ *entails* C, and write $(\mathcal{R}, \mathcal{T}) \models C$, if every model of $\mathcal{R}$ and $\mathcal{T}$ validates C. We say that C is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$ if some model of $\mathcal{R}$ and $\mathcal{T}$ satisfies $\{C\}$. Similarly, X is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$ if some model of $\mathcal{R}$ and $\mathcal{T}$ satisfies X. Observe that $(\mathcal{R}, \mathcal{T}) \models C$ iff $\neg C$ is unsatisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$.

For "tableaux" defined in the next section we use also the concept constructor $\forall^\dagger R.C$, where C is a concept without $\forall^\dagger$. A concept $\forall^\dagger R.C$ has the same semantics as $\forall R.C$ (i.e. $(\forall^\dagger R.C)^{\mathcal{I}} = (\forall R.C)^{\mathcal{I}}$) but its use in tableaux is more restricted due to $\widetilde{Sf}(\_)$ defined as follows. If C is not of the form $\forall^\dagger R.D$ then let $\widetilde{Sf}(C)$ be the set of all sub-concepts of C and sub-concepts of $\overline{C}$ including themselves, else let $\widetilde{Sf}(C) = \widetilde{Sf}(D)$. Let $\widetilde{Sf}(X) = \bigcup_{C \in X} \widetilde{Sf}(C) \cup \{\bot\}$ and let $\widetilde{Sf}_{\mathcal{R}}(X)$ be the set $\widetilde{Sf}(X) \cup \{\forall S.C, \exists S.\overline{C} \mid (\forall R.C) \in \widetilde{Sf}(X)$ and $S \sqsubseteq R \in \mathcal{R}$ for some $R\}$.

Note: We now assume that concepts are in *negation normal form*, where $\doteq$ and $\sqsubseteq$ are translated away and $\neg$ occurs only directly before atomic concepts, treating $\forall^\dagger$ as $\forall$. In $\mathcal{SHI}$, every concept C has a logically equivalent concept C' which is in negation normal form. We write $\overline{C}$ for the negation normal form of $\neg C$: thus, $\overline{\forall^\dagger R.D} = \overline{\forall R.D} = \exists R.\overline{D}$ and $\overline{\exists R.D} = \forall R.\overline{D}$.

3 A Tableau Calculus for $\mathcal{SHI}$

We consider tableaux w.r.t. $(\mathcal{R}, \mathcal{T})$, where $\mathcal{R}$ is a $\mathcal{SHI}$ RBox and $\mathcal{T}$ is a TBox. A *tableau rule* w.r.t. $(\mathcal{R}, \mathcal{T})$ consists of a numerator X and a (finite) list of denominators $Y_1, Y_2, \ldots, Y_k$, all of which are finite sets of concepts. Such a rule (ρ) is written in the form

$$(\rho)\ \frac{X}{Y_1 \mid \ldots \mid Y_k}$$

As we shall see later, each rule is read downwards as "if the numerator X is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$, then there exists a denominator Y_i which is also satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$". The numerator of each tableau rule contains one or more distinguished concepts called the *principal concepts*.

We write $X; Y$ for $X \cup Y$, and $X; C$ for $X \cup \{C\}$.

The tableau calculus $C\mathcal{SHI}$ for $\mathcal{SHI}$ is shown in Figure 2. The rules $(\sqcap)$, $(\sqcup)$, (H), $(H^\dagger)$, $(cut_\forall)$, (cut_B), and (cut_5) are *static* rules, while $(\exists R)$ is a *transitional rule*. Note that we include the principal concept of the static rules in their denominators. Thus, the numerator of any static rule is a subset of every one of

$$(\bot)\ \frac{X\ ;\ C\ ;\ \overline{C}}{\bot} \qquad (\sqcap)\ \frac{X\ ;\ C \sqcap D}{X\ ;\ C \sqcap D\ ;\ C\ ;\ D} \qquad (\sqcup)\ \frac{X\ ;\ C \sqcup D}{X\ ;\ C \sqcup D\ ;\ C \mid X\ ;\ C \sqcup D\ ;\ D}$$

$$(H)\ \frac{X\ ;\ \forall R.C}{X\ ;\ \forall R.C\ ;\ \forall S.C}\ \text{if } S \sqsubseteq R \in \mathcal{R} \qquad (H^\dagger)\ \frac{X\ ;\ \forall^\dagger R.C}{X\ ;\ \forall^\dagger R.C\ ;\ \forall^\dagger S.C}\ \text{if } S \sqsubseteq R \in \mathcal{R}$$

$$(cut_\forall)\ \frac{X}{X\ ;\ \forall R.C \mid X\ ;\ \exists R.\overline{C}}\ \text{if } \forall R.C \in \widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T})$$

$$(cut_B)\ \frac{X}{X\ ;\ C \mid X\ ;\ \overline{C}\ ;\ \forall^\dagger R^-.\exists R.\overline{C}}\ \text{if } \forall R.C \in \widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T})$$

$$(cut_5)\ \frac{X}{X\ ;\ \forall R.C \mid X\ ;\ \exists R.\overline{C}\ ;\ \forall^\dagger R^-.\exists R.\overline{C}}\ \begin{array}{l}\text{if } \forall R.C \in \widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T})\\ \text{and } R \circ R \sqsubseteq R \in \mathcal{R}\end{array}$$

$$(\exists R)\ \frac{X\ ;\ \exists R.C}{trans_{\mathcal{R}}(X,R)\ ;\ C\ ;\ \mathcal{T}}\ \text{where}$$

$$\begin{aligned} trans_{\mathcal{R}}(X,R) = {} & \{D \mid ((\forall S.D \in X) \vee (\forall^\dagger S.D \in X)) \wedge (R \sqsubseteq S \in \mathcal{R})\} \\ & \cup \{\forall S.D \in X \mid (R \sqsubseteq S \in \mathcal{R}) \wedge (S \circ S \sqsubseteq S \in \mathcal{R})\} \\ & \cup \{\forall^\dagger S.D \in X \mid (R \sqsubseteq S \in \mathcal{R}) \wedge (S \circ S \sqsubseteq S \in \mathcal{R})\} \end{aligned}$$

Fig. 2. Tableau Rules w.r.t. $(\mathcal{R}, \mathcal{T})$ for Calculus $\mathcal{CSHI}$

its denominators. The subscripts in (cut_B) and (cut_5) are names of the modal axioms $(B) : \varphi \to \Box\Diamond\varphi$ and $(5) : \Diamond\varphi \to \Box\Diamond\varphi$, from which they are derived.

A set X is *closed* w.r.t. a tableau rule if applying that rule to X gives back X as one of the denominators. *We implicitly assume that a static rule is applied to X only when X is not closed w.r.t. that rule and treat this as an (additional) condition for applying the rule.* Consequently, the rules (H) and $(H^\dagger)$ require that $S \neq R$ and the static rules with two denominators are applicable only if *both denominators are different from their numerator.*

A *$\mathcal{CSHI}$-tableau* (tableau, for short) for X w.r.t. $(\mathcal{R}, \mathcal{T})$ is a tree with root $(X; \mathcal{T})$ whose nodes carry finite sets of concepts obtained from their parent nodes by instantiating a $\mathcal{CSHI}$-tableau rule w.r.t. $(\mathcal{R}, \mathcal{T})$ with the proviso that: if a child s carries a set Y and no rule is applicable to Y or Y has already appeared on the branch from the root to s then s is an *end node.*

Remark 1. Note that the rules guarantee that $\mathcal{T}$ is contained in every world of the model under construction to ensure that $\mathcal{T}$ is globally satisfied as required, and to allow a simpler presentation of the rules.

A branch in a tableau is *closed* if its end node carries only $\bot$. A tableau is *closed* if every one of its branches is closed. A tableau is *open* if it is not closed.

A finite set X of concepts is *consistent* w.r.t. $(\mathcal{R}, \mathcal{T})$ if every tableau for X w.r.t. $(\mathcal{R}, \mathcal{T})$ is open. If some tableau for X w.r.t. $(\mathcal{R}, \mathcal{T})$ is closed then X is *inconsistent* w.r.t. $(\mathcal{R}, \mathcal{T})$.

Calculus $\mathcal{CSHI}$ is *sound* if for every $\mathcal{SHI}$ RBox $\mathcal{R}$, every TBox $\mathcal{T}$, and every finite set X of concepts, X is satisfiable w.r.t. $(\mathcal{R},\mathcal{T})$ implies X is consistent w.r.t. $(\mathcal{R},\mathcal{T})$. It is *complete* if for every $\mathcal{SHI}$ RBox $\mathcal{R}$, every TBox $\mathcal{T}$, and every finite set X of concepts, X is consistent w.r.t. $(\mathcal{R},\mathcal{T})$ implies X is satisfiable w.r.t. $(\mathcal{R},\mathcal{T})$.

Example 1. Let $\mathcal{R}$ be the smallest RBox satisfying the conditions adopted for RBoxes and containing $S \circ S \sqsubseteq S$ and $S \sqsubseteq R$. We give below a closed $\mathcal{CSHI}$ tableau for $C \sqcap \exists S.\exists S.\forall R^-.\overline{C}$ w.r.t. $(\mathcal{R},\emptyset)$, in which a superscript $*$ marks the principal concept of a rule application except the cut rules. As shown later, $\mathcal{CSHI}$ is sound, hence every model of $\mathcal{R}$ validates $C \sqsubseteq \forall S.\forall S.\exists R^-.C$.

$$\dfrac{\dfrac{\dfrac{C \sqcap^* \exists S.\exists S.\forall R^-.\overline{C}}{C;\ \exists S.\exists S.\forall R^-.\overline{C}}}{\dfrac{C;\ \exists S.\exists S.\forall R^-.\overline{C};\overline{C}}{\bot}} \quad \dfrac{\dfrac{\dfrac{\dfrac{C;\ \exists S.\exists S.\forall R^-.\overline{C};\ C;\ \forall^{\dagger *}R.\exists R^-.C}{C;\ \exists^* S.\exists S.\forall R^-.\overline{C};\ C;\ \forall^\dagger R.\exists R^-.C;\ \forall^\dagger S.\exists R^-.C}}{\exists^* S.\forall R^-.\overline{C};\ \exists R^-.C;\ \forall^\dagger S.\exists R^-.C}}{\forall R^-.\overline{C};\ \exists R^-.C;\ \forall^\dagger S.\exists R^-.C}}{\bot}$$

A tableau rule w.r.t. $(\mathcal{R},\mathcal{T})$ is *sound* if it has the property that if the numerator is satisfiable w.r.t. $(\mathcal{R},\mathcal{T})$ then so is one of the denominators.

Lemma 1. *The calculus $\mathcal{CSHI}$ is sound.*

Proof. The calculus $\mathcal{CSHI}$ is sound because all rules of $\mathcal{CSHI}$ are sound.

Let the closure $\mathrm{Cl}(\mathcal{R},\mathcal{T},X)$ be the set

$$\{D, \forall^\dagger R.D \mid D \in \widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T}) \text{ and } R \text{ is a } \mathcal{SHI}\text{-role appearing in } (\mathcal{R},\mathcal{T},X)\}.$$

Let n be the size of $(\mathcal{R},\mathcal{T},X)$, i.e. the sum of the lengths of the concepts of $X \cup \mathcal{T}$ and the lengths of the assertions of $\mathcal{R}$. Then the size of $\widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T})$ is $O(n^2)$ and the size of $\mathrm{Cl}(\mathcal{R},\mathcal{T},X)$ is $O(n^3)$. Note that if a concept C appears in a tableau for X w.r.t. $(\mathcal{R},\mathcal{T})$ then $C \in \mathrm{Cl}(\mathcal{R},\mathcal{T},X)$: calculus $\mathcal{CSHI}$ thus has the *analytic superformula property* [5].

4 Completeness

4.1 Proving Completeness Via Model Graphs

We prove completeness of our calculus via model graphs following [11,5,10]. A model graph is a tuple $\langle \Delta, \tau, \mathcal{C}, \mathcal{E}\rangle$, where: Δ is a finite set; τ is a distinguished element of Δ; $\mathcal{C}$ is a function that maps each element of Δ to a set of concepts; and $\mathcal{E}$ is a function that maps each role name to a binary relation on Δ. A model graph $\langle \Delta, \tau, \mathcal{C}, \mathcal{E}\rangle$ is *saturated* if every $x \in \Delta$ satisfies:

1. if $C \sqcap D \in \mathcal{C}(x)$ then $\{C, D\} \subseteq \mathcal{C}(x)$,
2. if $C \sqcup D \in \mathcal{C}(x)$ then $C \in \mathcal{C}(x)$ or $D \in \mathcal{C}(x)$,

3. if $(\forall R.C \in \mathcal{C}(x)$ or $\forall^\dagger R.C \in \mathcal{C}(x))$ and $\mathcal{E}(R)(x,y)$ holds then $C \in \mathcal{C}(y)$,
4. if $\exists R.C \in \mathcal{C}(x)$ then there exists $y \in \Delta$ with $\mathcal{E}(R)(x,y)$ and $C \in \mathcal{C}(y)$.

A saturated model graph $\langle \Delta, \tau, \mathcal{C}, \mathcal{E} \rangle$ is *consistent* if no $x \in \Delta$ has a $\mathcal{C}(x)$ containing $\bot$ or containing a pair C, $\overline{C}$ for some concept C. Given a model graph $M = \langle \Delta, \tau, \mathcal{C}, \mathcal{E} \rangle$, the *interpretation corresponding to* M is the interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ where $A^{\mathcal{I}} = \{x \in \Delta \mid A \in \mathcal{C}(x)\}$ for every atomic concept A and $R^{\mathcal{I}} = \mathcal{E}(R)$ for every role name R.

Lemma 2. *If $\mathcal{I}$ is the interpretation corresponding to a consistent saturated model graph $\langle \Delta, \tau, \mathcal{C}, \mathcal{E} \rangle$, then for every $x \in \Delta$ and $C \in \mathcal{C}(x)$ we have $x \in C^{\mathcal{I}}$.*

Proof. By induction on the structure of C.

Given a $\mathcal{SHI}$ RBox $\mathcal{R}$, a TBox $\mathcal{T}$, and a finite set X of concepts consistent w.r.t. $(\mathcal{R}, \mathcal{T})$, we construct a model of $\mathcal{R}$ and $\mathcal{T}$ that satisfies X by constructing a consistent saturated model graph $\langle \Delta, \tau, \mathcal{C}, \mathcal{E} \rangle$ such that the corresponding interpretation is a model of $\mathcal{R}$, $X \subseteq \mathcal{C}(\tau)$, and $\mathcal{T} \subseteq \mathcal{C}(x)$ for every $x \in \Delta$.

4.2 Saturation

For a finite set $X \supseteq \mathcal{T}$ of concepts that is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$, a set Y of concepts is called a *saturation of* X w.r.t. $(\mathcal{R}, \mathcal{T})$ if Y is a maximal set consistent w.r.t. $(\mathcal{R}, \mathcal{T})$ that is obtainable from X (as a leaf node) by applications of the static rules.

Lemma 3. *Let X be a finite set of concepts consistent w.r.t. $(\mathcal{R}, \mathcal{T})$, and Y a saturation of X w.r.t. $(\mathcal{R}, \mathcal{T})$. Then $X \subseteq Y \subseteq \mathrm{Cl}(\mathcal{R}, \mathcal{T}, X)$ and Y is closed w.r.t. the static rules. Furthermore, there is an effective procedure that constructs such a set Y from X and $(\mathcal{R}, \mathcal{T})$.*

Proof. It is clear that $X \subseteq Y \subseteq \mathrm{Cl}(\mathcal{R}, \mathcal{T}, X)$. Observe that if a static rule is applicable to Y, then one of the corresponding instances of the denominators is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$. Since Y is a saturation of X w.r.t. $(\mathcal{R}, \mathcal{T})$, it is closed w.r.t. the static rules.

We construct a saturation of X w.r.t. $(\mathcal{R}, \mathcal{T})$ as follows: let $Y := X$; while a static rule is applicable to Y and has a corresponding denominator instance Z which is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$ and strictly contains Y, set $Y := Z$. At each iteration, $Y \subset Z \subseteq \mathrm{Cl}(\mathcal{R}, \mathcal{T}, X)$, so this process always terminates. Clearly, the resulting set Y is a saturation of X w.r.t. $(\mathcal{R}, \mathcal{T})$.

4.3 Constructing Model Graphs

Figure 3 contains an algorithm for constructing a model graph. Algorithm 1 assumes that X is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$ and constructs a model of $\mathcal{R}$ and $\mathcal{T}$ that satisfies X. Algorithm 1 always terminates because each $x \in \Delta$ has a unique finite set $\mathcal{C}(x)$, which is a subset of the finite set $\mathrm{Cl}(\mathcal{R}, \mathcal{T}, X)$, so eventually Step 2(a)ii always finds a proxy.

Algorithm 1
Input: a $\mathcal{SHI}$ RBox $\mathcal{R}$, a TBox $\mathcal{T}$, and a finite set X of concepts,
where X is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$.
Output: a model graph $M = \langle \Delta, \tau, \mathcal{C}, \mathcal{E} \rangle$.

1. For an arbitrary node name τ, let $\Delta := \{\tau\}$, and $\mathcal{E}'(R) := \emptyset$ for every role name R. Let $\mathcal{C}(\tau)$ be a saturation of $X \cup \mathcal{T}$ w.r.t. $(\mathcal{R}, \mathcal{T})$ and mark τ as unexpanded.
2. While Δ contains unexpanded elements, take one, say x, and do:
   (a) For every concept $\exists R.C \in \mathcal{C}(x)$:
      i. Let $Y = trans_{\mathcal{R}}(\mathcal{C}(x), R) \cup \{C\} \cup \mathcal{T}$ be the result of applying rule $(\exists R)$ to $\mathcal{C}(x)$, and let Z be a saturation of Y w.r.t. $(\mathcal{R}, \mathcal{T})$.
      ii. If there is a (proxy) $y \in \Delta$ with $\mathcal{C}(y) = Z$ then add pair (x, y) to $\mathcal{E}'(R)$;
      iii. Else add a new element y with $\mathcal{C}(y) := Z$ to Δ, mark y as unexpanded, and add the pair (x, y) to $\mathcal{E}'(R)$.
   (b) Mark x as expanded.
3. Let $\mathcal{E}$ be the least extension of $\mathcal{E}'$ that satisfies the conditions $\mathcal{E}(R^-) = (\mathcal{E}(R))^{-1}$ for every role name R and the assertions of $\mathcal{R}$.

Fig. 3. Constructing a Model Graph (Using Limited Caching)

Lemma 4. *Let $\mathcal{R}$ be a $\mathcal{SHI}$ RBox, $\mathcal{T}$ a TBox, and X a finite set of concepts consistent w.r.t. $(\mathcal{R}, \mathcal{T})$. Let $M = \langle \Delta, \tau, \mathcal{C}, \mathcal{E} \rangle$ be the model graph constructed by Algorithm 1 for $(\mathcal{R}, \mathcal{T}, X)$, and $\mathcal{I}$ be the interpretation corresponding to M. Then $\mathcal{I}$ is a model of $\mathcal{R}$ and $\mathcal{T}$ that satisfies X.*

Proof. We first show that M is a consistent saturated model graph. It is sufficient to show that for every $x, y \in \Delta$, if $(\forall R.C \in \mathcal{C}(x)$ or $\forall^\dagger R.C \in \mathcal{C}(x))$ and $\mathcal{E}(R)(x, y)$ holds then $C \in \mathcal{C}(y)$. We prove the following stronger assertion:

> *For all $x, y \in \Delta$ and all $\mathcal{SHI}$-roles R and S: if $\forall R.C \in \mathcal{C}(x)$ (resp. $\forall^\dagger R.C \in \mathcal{C}(x)$) and $\mathcal{E}(S)(x, y)$ and $S \sqsubseteq R \in \mathcal{R}$ then (i) $C \in \mathcal{C}(y)$, and (ii) $\forall R.C \in \mathcal{C}(y)$ (resp. $\forall^\dagger R.C \in \mathcal{C}(y)$) if $R \circ R \sqsubseteq R \in \mathcal{R}$.*

We prove this assertion by induction on the number of steps needed to derive $\mathcal{E}(S)(x, y)$ during Step 3 of Algorithm 1. Suppose that $\forall R.C \in \mathcal{C}(x)$ (resp. $\forall^\dagger R.C \in \mathcal{C}(x)$), $S \sqsubseteq R \in \mathcal{R}$, and $\mathcal{E}(S)(x, y)$ holds. There are the following cases to consider:

1. The assertion trivially holds for the base case when $\mathcal{E}'(S)(x, y)$ holds. Thus we created y as an S-successor for x in order to fulfil some eventuality $\exists S.D \in \mathcal{C}(x)$. But when constructing y, the rule $(\exists R)$ puts $C \in \mathcal{C}(y)$ since $S \sqsubseteq R$ and $\forall R.C \in \mathcal{C}(x)$ (resp. $\forall^\dagger R.C \in \mathcal{C}(x)$). Similarly, it puts $\forall R.C \in \mathcal{C}(y)$ (resp. $\forall^\dagger R.C \in \mathcal{C}(y)$) if $R \circ R \sqsubseteq R \in \mathcal{R}$.
2. Case $\mathcal{E}(S)(x, y)$ is derived from $\mathcal{E}(S')(x, y)$ and $S' \sqsubseteq S \in \mathcal{R}$. Since $S \sqsubseteq R \in \mathcal{R}$ by assumption, we have that $S' \sqsubseteq R \in \mathcal{R}$ by RBox 4. The derivation of $\mathcal{E}(S')(x, y)$ is shorter by one step, so it falls under the induction hypothesis (by putting S' for S in the statement of the assertion). The desired conclusions follow immediately.

3. Case $\mathcal{E}(S)(x,y)$ is derived from $\mathcal{E}(S)(x,z)$, $\mathcal{E}(S)(z,y)$, and $S \circ S \sqsubseteq S \in \mathcal{R}$. Assume that $\forall R.C \in \mathcal{C}(x)$ (the case $\forall^{\dagger} R.C \in \mathcal{C}(x)$ is similar). Since $S \sqsubseteq R$ and $\forall R.C \in \mathcal{C}(x)$, the rule (H) would have put $\forall S.C \in \mathcal{C}(x)$. Applying the inductive hypothesis to $\mathcal{E}(S)(x,z)$ (and putting S for R in the statement of the assertion), we obtain that $\forall S.C \in \mathcal{C}(z)$ since $S \circ S \sqsubseteq S \in \mathcal{R}$. Applying the inductive hypothesis to $\mathcal{E}(S)(z,y)$ (and putting S for R in the statement of the assertion), we obtain that $C \in \mathcal{C}(y)$. If $R \circ R \sqsubseteq R \in \mathcal{R}$ then, by the inductive assumption, $\forall R.C \in \mathcal{C}(z)$, and $\forall R.C \in \mathcal{C}(y)$.
4. Case $\mathcal{E}(S)(x,y)$ is derived from $\mathcal{E}(S^-)(y,x)$. There are two subcases:
   (a) Case $\forall R.C \in \mathcal{C}(x)$. By the (H) rule, $\forall S.C \in \mathcal{C}(x)$. If $C \notin \mathcal{C}(y)$ then, by the (cut_B) rule, $\overline{C} \in \mathcal{C}(y)$ and $\forall^{\dagger} S^-.\exists S.\overline{C} \in \mathcal{C}(y)$. Applying the inductive assumption to $\mathcal{E}(S^-)(y,x)$ (and putting S^- for both S and R in the statement of the assertion) gives us that $\exists S.\overline{C} \in \mathcal{C}(x)$. But $\{\exists S.\overline{C}, \forall S.C\}$ is an inconsistent subset of $\mathcal{C}(x)$, contradicting the consistency of $\mathcal{C}(x)$, hence $C \in \mathcal{C}(y)$. If $R \circ R \sqsubseteq R \in \mathcal{R}$ and $\forall R.C \notin \mathcal{C}(y)$ then, by the (cut_5) rule, $\exists R.\overline{C} \in \mathcal{C}(y)$ and $\forall^{\dagger} R^-.\exists R.\overline{C} \in \mathcal{C}(y)$, and by the inductive assumption, together with $\mathcal{E}(S^-)(y,x)$ and $S^- \sqsubseteq R^- \in \mathcal{R}$ it implies that $\exists R.\overline{C} \in \mathcal{C}(x)$, which contradicts $\forall R.C \in \mathcal{C}(x)$.
   (b) Case $\forall^{\dagger} R.C \in \mathcal{C}(x)$. There are two subcases since we can create such a $\forall^{\dagger} R$-concept using only the rules (cut_B) and (cut_5):
   (cut_B): Then $C = \exists R^-.\overline{D}$ with $\overline{D} \in \mathcal{C}(x)$, and $\forall R^-.D \in \widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T})$. Suppose that $C \notin \mathcal{C}(y)$, i.e. $\exists R^-.\overline{D} \notin \mathcal{C}(y)$. Then, by the rule $(cut_{\forall})$, $\forall R^-.D \in \mathcal{C}(y)$. By RBox 2, we have $S^- \sqsubseteq R^- \in \mathcal{R}$ from $S \sqsubseteq R \in \mathcal{R}$. Since $\forall R^-.D \in \mathcal{C}(y)$ and $S^- \sqsubseteq R^- \in \mathcal{R}$ and $\mathcal{E}(S^-)(y,x)$, by the inductive assumption, $D \in \mathcal{C}(x)$, which contradicts $\overline{D} \in \mathcal{C}(x)$. Therefore $C \in \mathcal{C}(y)$.
   Suppose that $R \circ R \sqsubseteq R \in \mathcal{R}$ and $\forall^{\dagger} R.C \notin \mathcal{C}(y)$, i.e. $\forall^{\dagger} R.\exists R^-.\overline{D} \notin \mathcal{C}(y)$. By RBox 3, we have $R^- \circ R^- \sqsubseteq R^- \in \mathcal{R}$ too. Then, by the (cut_5) rule, $\forall R^-.D \in \mathcal{C}(y)$. Analogously as for the previous paragraph, it follows that $D \in \mathcal{C}(x)$, which contradicts $\overline{D} \in \mathcal{C}(x)$. Therefore, $R \circ R \sqsubseteq R \in \mathcal{R}$ implies $\forall^{\dagger} R.C \in \mathcal{C}(y)$.
   (cut_5): Then $C = \exists R^-.\overline{D}$ with $\exists R^-.\overline{D} \in \mathcal{C}(x)$, $\forall R^-.D \in \widetilde{Sf}_{\mathcal{R}}(X \cup \mathcal{T})$ and $R^- \circ R^- \sqsubseteq R^- \in \mathcal{R}$.
   Suppose $C \notin \mathcal{C}(y)$, i.e. $\exists R^-.\overline{D} \notin \mathcal{C}(y)$. Then $\forall R^-.D \in \mathcal{C}(y)$ by the rule $(cut_{\forall})$. By the inductive hypothesis, it follows that $\forall R^-.D \in \mathcal{C}(x)$, which contradicts $\exists R^-.\overline{D} \in \mathcal{C}(x)$. Therefore $C \in \mathcal{C}(y)$.
   Suppose $\forall^{\dagger} R.C \notin \mathcal{C}(y)$, i.e. $\forall^{\dagger} R.\exists R^-.\overline{D} \notin \mathcal{C}(y)$. Then $\forall R^-.D \in \mathcal{C}(y)$ by the (cut_5) rule. By the inductive assumption, it follows that $\forall R^-.D \in \mathcal{C}(x)$, which contradicts $\exists R^-.\overline{D} \in \mathcal{C}(x)$. Therefore $\forall^{\dagger} R.C \in \mathcal{C}(y)$.

By the construction of M, the corresponding interpretation $\mathcal{I}$ is a model of $\mathcal{R}$, and we have that $X \subseteq \mathcal{C}(\tau)$ and $\mathcal{T} \subseteq \mathcal{C}(x)$ for every $x \in \Delta$. Hence, by Lemma 2, $\mathcal{I}$ is a model of $\mathcal{R}$ and $\mathcal{T}$ that satisfies X.

The following theorem immediately follows from Lemmas 1 and 4.

Theorem 1. *The calculus CSHI is sound and complete.*

5 A Simple EXPTIME Decision Procedure for $\mathcal{SHI}$

The naive decision procedure for $\mathcal{SHI}$ that explores tableaux in the usual way has co-2NEXPTIME complexity because each branch in a tableau may have an exponential length, meaning that the whole tableau (tree) may have a double-exponential number of nodes. Algorithm 1 has the same complexity because of the computation of saturations (using naive tableaux).

By simulating the creation of saturations and checking whether the resulting model graph is consistent, it is easy to alter Algorithm 1 so that it explicitly checks, rather than assumes, that X is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$. Simulating the creation of (a *candidate* for) a saturation is done nondeterministically in linear time. Since $\mathrm{Cl}(\mathcal{R}, \mathcal{T}, X)$ has size $O(n^3)$, the constructed model graph has size $2^{O(n^3)}$. So the resulting algorithm for checking whether X is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$, where $\mathcal{R}$ is a $\mathcal{SHI}$ RBox and $\mathcal{T}$ is a TBox $\mathcal{T}$, runs in NEXPTIME.

In Figure 4 we present an EXPTIME decision procedure for $\mathcal{SHI}$ which directly uses the tableau rules of $C\mathcal{SHI}$ to create an and-or graph as follows.

A node in the constructed and-or graph is a record with three attributes:

content: the set of concepts carried by the node
status: {`unexpanded`, `expanded`, `sat`, `unsat`}
kind: {`and-node`, `or-node`}

To check whether a given finite set X is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$, where $\mathcal{R}$ is a $\mathcal{SHI}$ RBox and $\mathcal{T}$ is a TBox, the content of the initial node τ with status `unexpanded` is $X \cup \mathcal{T}$. The main while-loop continues processing nodes until the status of τ is determined to be in {`sat`, `unsat`}, or until every node is expanded, whichever happens first.

Inside the main loop, Steps (2b) to (2f) try to apply one and only one of the tableau rules in the order $(\bot)$, $(\sqcap)$, (H), $(H^\dagger)$, $(\sqcup)$, $(cut_\forall)$, (cut_B), (cut_5), $(\exists R)$ to the current node v. The set $\mathcal{D}$ contains the contents of the resulting denominators of v. If the applied tableau rule is $(\sqcap)$ or (H) or $(H^\dagger)$ then v has one denominator in $\mathcal{D}$; if the applied rule is $(\sqcup)$ or $(cut_\forall)$ or (cut_B) or (cut_5) then v has two denominators in $\mathcal{D}$; otherwise, each concept $\exists R.C \in v.content$ contributes one appropriate denominator to $\mathcal{D}$. At Step (2g), for every denominator in $\mathcal{D}$, we create the required successor in the graph G only if it does not yet exist in the graph: this step merely mimics Algorithm 1 and therefore uses global caching.

In Algorithm 2, a node that contains both C and $\overline{C}$ for some concept C becomes an end-node with status `unsat` (i.e. unsatisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$). A node to which no tableau rule is applicable becomes an end-node with status `sat` (i.e. satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$).

On the other hand, an application of $(\sqcup)$ or $(cut_\forall)$ or (cut_B) or (cut_5) to a node v causes v to be an *or-node*, while an application of $(\sqcap)$ or (H) or $(H^\dagger)$ or $(\exists R)$ to a node v causes v to be an *and-node*. Steps (2h) and (2i) try to compute the status of such a non-end-node v using the kind (or-node/and-node) of v and the status of the successors of v, treating `unsat` as irrevocably `false` and `sat` as irrevocably `true`.

Algorithm 2
Input: a $\mathcal{SHI}$ RBox $\mathcal{R}$, a TBox $\mathcal{T}$, and a finite set X of concepts
Output: an and-or graph $G = \langle V, E \rangle$ with $\tau \in V$ as the initial node such that $\tau.status = \texttt{sat}$ iff X is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$

1. create a new node τ with $\tau.content := X \cup \mathcal{T}$ and $\tau.status := \texttt{unexpanded}$; let $V := \{\tau\}$ and $E := \emptyset$;
2. while $\tau.status \notin \{\texttt{sat}, \texttt{unsat}\}$ and we can choose an unexpanded node $v \in V$ do:
   (a) $\mathcal{D} := \emptyset$;
   (b) if no $\mathcal{CSHI}$-tableau rule is applicable to $v.content$ then $v.status := \texttt{sat}$
   (c) else if $(\bot)$ is applicable to $v.content$ then $v.status := \texttt{unsat}$
   (d) else if $(\sqcap)$ or (H) or $(H^\dagger)$ is applicable to $v.content$ giving denominator Y then $v.kind := \texttt{and-node}$, $\mathcal{D} := \{Y\}$
   (e) else if $(\sqcup)$ or $(cut_\forall)$ or (cut_B) or (cut_5) is applicable to $v.content$ giving denominators Y_1 and Y_2 (both different from $v.content$) then $v.kind := \texttt{or-node}$, $\mathcal{D} := \{Y_1, Y_2\}$
   (f) else
      i. $v.kind := \texttt{and-node}$,
      ii. for every $\exists R.C \in v.content$, apply $(\exists R)$ to $v.content$ giving denominator $trans_\mathcal{R}(v.content, R) \cup \{C\} \cup \mathcal{T}$ and add this denominator to $\mathcal{D}$;
   (g) for every denominator $Y \in \mathcal{D}$ do
      i. if some (proxy) $w \in V$ has $w.content = Y$ then add edge (v, w) to E
      ii. else let w be a new node, set $w.content := Y$, $w.status := \texttt{unexpanded}$, add w to V, and add edge (v, w) to E;
   (h) if ($v.kind = \texttt{or-node}$ and one of the successors of v has status $\texttt{sat}$) or ($v.kind = \texttt{and-node}$ and all the successors of v have status $\texttt{sat}$) then $v.status := \texttt{sat}$, $propagate(G, v)$
   (i) else if ($v.kind = \texttt{and-node}$ and one of the successors of v has status $\texttt{unsat}$) or ($v.kind = \texttt{or-node}$ and all the successors of v have status $\texttt{unsat}$) then $v.status := \texttt{unsat}$, $propagate(G, v)$
   (j) else $v.status := \texttt{expanded}$;
3. if $\tau.status \notin \{\texttt{sat}, \texttt{unsat}\}$ then for every node $v \in V$ with $v.status \neq \texttt{unsat}$, set $v.status := \texttt{sat}$;

Fig. 4. A Simple EXPTIME Decision Procedure for $\mathcal{SHI}$

If these steps cannot determine the status of v as `sat` or `unsat`, then its status is set to `expanded`. But if these steps do determine the status of a node v to be `sat` or `unsat`, this information is itself propagated to the predecessors of v in the and-or graph G via the routine $propagate(G, v)$, explained shortly.

The main loop ends when the status of the initial node τ becomes `sat` or `unsat` or all nodes of the graph have been expanded. In the latter case, all nodes with status $\neq$ `unsat` are given status `sat` (effectively giving the status *open* to tableau branches which loop).

The procedure *propagate* used in the above algorithm is specified in Figure 5. As parameters, it accepts an and-or graph G and a node v with (irrevocable)

Procedure *propagate*(G, v)
Parameters: an and-or graph $G = \langle V, E\rangle$ and $v \in V$ with $v.status \in \{\texttt{sat}, \texttt{unsat}\}$
Returns: a modified and-or graph $G = \langle V, E\rangle$

1. $queue := \{v\}$;
2. while $queue$ is not empty do
3. (a) extract x from $queue$;
   (b) for every $u \in V$ with $(u, x) \in E$ and $u.status = \texttt{expanded}$ do
      i. if ($u.kind = \texttt{or-node}$ and one of the successors of u has status $\texttt{sat}$)
         or ($u.kind = \texttt{and-node}$ and all the successors of u have status $\texttt{sat}$) then
            $u.status := \texttt{sat}$, $queue := queue \cup \{u\}$
      ii. else if ($u.kind = \texttt{and-node}$ and one of the successors of u has status $\texttt{unsat}$)
         or ($u.kind = \texttt{or-node}$ and all the successors of u have status $\texttt{unsat}$) then
            $u.status := \texttt{unsat}$, $queue := queue \cup \{u\}$;

Fig. 5. Propagating Satisfiability and Unsatisfiability Through an And-Or Graph

status $\texttt{sat}$ or $\texttt{unsat}$. The purpose is to propagate the status of v through the and-or graph and alter G to reflect the new information.

Initially, the queue of nodes to be processed contains only v. While the queue is not empty: a node x is extracted; the status of x is propagated to each predecessor u of x in an appropriate way; and if the status of u becomes (irrevocably) $\texttt{sat}$ or $\texttt{unsat}$ then u is inserted into the queue for further propagation.

This construction thus uses both caching and propagation techniques.

Proposition 1. *Algorithm 2 runs in EXPTIME.*

Proof. Let $G = \langle V, E\rangle$ be the graph constructed by Algorithm 2 for $(\mathcal{R}, \mathcal{T}, X)$ and n be the size of the input, i.e. the sum of the lengths of the concepts of $X \cup \mathcal{T}$ and the lengths of the assertions of $\mathcal{R}$.

For every $v \in V$, we have that $v.content \subseteq \mathrm{Cl}(\mathcal{R}, \mathcal{T}, X)$, hence the size of $v.content$ is $O(n^3)$. For every $v, v' \in V$, if $v \neq v'$ then $v.content \neq v'.content$. Hence V contains $2^{O(n^3)}$ nodes. Every $v \in V$ is expanded (by Steps (2a)–(2j)) only once and such a task takes $2^{O(n^3)}$ time units without counting the execution time of the procedure *propagate*. When $v.status$ becomes $\texttt{sat}$ or $\texttt{unsat}$, the procedure *propagate* executes $2^{O(n^3)}$ basic steps directly involved with v. Hence the total time of the executions of *propagate* is of rank $2^{O(n^3)}$. The time complexity of Algorithm 2 is therefore of rank $2^{O(n^3)}$.

Lemma 5. *It is an invariant of Algorithm 2 that for every $v \in V$:*

1. *if $v.status = \texttt{unsat}$ then*
   - *$v.content$ contains both C and $\overline{C}$ for some concept C,*
   - *or $v.kind = \texttt{and-node}$ and there exists $(v, w) \in E$ such that $w \neq v$ and $w.status = \texttt{unsat}$,*
   - *or $v.kind = \texttt{or-node}$ and for every $(v, w) \in E$, $w.status = \texttt{unsat}$;*

2. *if* $v.status = \texttt{sat}$ *then*
   - *no* $\mathcal{CSHI}$*-tableau rule is applicable to* $v.content$*,*
   - *or* $v.kind = \texttt{or-node}$ *and there exists* $(v, w) \in E$ *with* $w.status = \texttt{sat}$*,*
   - *or* $v.kind = \texttt{and-node}$ *and for every* $(v, w) \in E$*,* $w.status = \texttt{sat}$*.*

(Since a static rule is applied to X *only when* X *is not closed w.r.t. the rule, if* $v.kind = \texttt{or-node}$ *and* $(v, w) \in E$ *then* $w \neq v$ *since* $w.content \neq v.content$*.)*

Proof. Lemma 5(1) clearly holds since these are the only three ways for a node to get status `unsat`. For Lemma 5(2) there is the possibility that the node gets status `sat` via Step 3 of Algorithm 2.

For a contradiction, assume that $v.status$ becomes `sat` because of Step 3 of Algorithm 2 and that all three clauses of the "then" part of Lemma 5(2) fail:

1. First, the rule assumed to be applicable to $v.content$ cannot be the $(\bot)$-rule as this would have put $v.status = \texttt{unsat}$, contradicting our assumption that $v.status = \texttt{sat}$. Thus the rule must be one of the remaining rules, meaning that $v.kind = \texttt{or-node}$ or $v.kind = \texttt{and-node}$ after this rule application.
2. Second, if $v.kind = \texttt{or-node}$ then v must have two successors created by one of the rules $(\sqcup)$, $(cut_\forall)$, (cut_B), (cut_5), since this is the only way for a node to have $v.kind = \texttt{or-node}$. If neither successor has status `sat` then they must both have status `unsat`. But Algorithm 2 and procedure *propagate* always ensure that `unsat` is propagated whenever it is found. As soon as the `unsat` status of the latter of these two children is found, the ensuing call to *propagate* would have ensured that $v.status = \texttt{unsat}$, contradicting our assumption that $v.status = \texttt{sat}$.
3. Third, if $v.kind = \texttt{and-node}$ then v has at least one successor w (say) with $(v, w) \in E$. If $w.status \neq \texttt{sat}$, then we must have $w.status = \texttt{unsat}$. But again, as soon as w gets status `unsat`, procedure *propagate* would ensure that $v.status = \texttt{unsat}$ too, contradicting our assumption that $v.status = \texttt{sat}$.

Lemma 6. *Let* $G = \langle V, E\rangle$ *be the graph constructed by Algorithm 2 for* $(\mathcal{R}, \mathcal{T}, X)$*. For every* $v \in V$*, if* $v.status = \texttt{unsat}$ *then* $v.content$ *is inconsistent w.r.t.* $(\mathcal{R}, \mathcal{T})$*.*

Proof. Using Lemma 5, we can construct a closed tableau w.r.t. $(\mathcal{R}, \mathcal{T})$ for $v.content$ by induction on the way v depends on its successors and by copying nodes so that the resulting structure is a (tree) tableau rather than a graph.

Let $G = \langle V, E\rangle$ be the graph constructed by Algorithm 2 for $(\mathcal{R}, \mathcal{T}, X)$. For $v \in V$ with $v.status = \texttt{sat}$, we say that $v_0 = v, v_1, \ldots, v_k$ with $k \geq 0$ is a *saturation path of* v *in* G if for each $1 \leq i \leq k$, we have $v_i.status = \texttt{sat}$, the edge $E(v_{i-1}, v_i)$ was created by an application of a static rule, and $v_k.content$ is closed w.r.t. the static rules. Observe that if $v_0, \ldots, v_k$ is a saturation path of v_0 in G then $v_0.content \subseteq \ldots \subseteq v_k.content$.

By Lemma 5, if $v.status = \texttt{sat}$ then there exists a saturation path of v in G.

Lemma 7. *Let* $G = \langle V, E\rangle$ *be the graph constructed by Algorithm 2 for* $(\mathcal{R}, \mathcal{T}, X)$*. If* $\tau.status = \texttt{sat}$ *then every tableau for* X *w.r.t.* $(\mathcal{R}, \mathcal{T})$ *is open.*

Proof. Let T be an arbitrary tableau for X w.r.t. $(\mathcal{R}, \mathcal{T})$. We maintain a *current node cn* of T that will follow edges of T to pin-point an open branch of T. Initially we set *cn* to be the root of T. We also keep a (finite) saturation path σ of the form $\sigma_0, \ldots, \sigma_k$ for some $\sigma_0 \in V$ and call σ the *current saturation path in* G. At the beginning, set $\sigma_0 := \tau$ and let σ be a saturation path for σ_0 in G: such a saturation path exists since $\tau.status = \texttt{sat}$. We maintain the invariant $cn.content \subseteq \sigma_k.content$, where $cn.content$ is the set carried by *cn*.

Remark 2. By the definition of saturation path, $\sigma_k.status = \texttt{sat}$. The invariant thus implies that the rule $(\bot)$ is not applicable to *cn*.

Clearly, the invariant holds at the beginning since $\sigma_0 = \tau$ and $\tau.content = cn.content$ and $\sigma_0.content \subseteq \sigma_k.content$. Depending upon the rule applied to *cn* in the tableau T, we maintain the invariant by changing the value of the current node *cn* of T and possibly also the current saturation path σ in G:

1. Case the tableau rule applied to *cn* is a static rule. Since $cn.content \subseteq \sigma_k.content$ and $\sigma_k.content$ is closed w.r.t. the static rules, *cn* has a successor u in T with $u.content \subseteq \sigma_k.content$. By setting $cn := u$, the invariant is maintained without changing σ.
2. Case the tableau rule applied to *cn* is the transitional rule $(\exists R)$ with principal concept $\exists R.D$, and the successor is $u \in T$.

   By the invariant, $\exists R.D \in \sigma_k.content$. So there must be a node $w \in V$ such that the edge $E(\sigma_k, w)$ was created by the application of $(\exists R)$ to $\sigma_k.content$ with $\exists R.D$ as the principal concept. Thus, $u.content \subseteq w.content$ by the form of the $(\exists R)$-rule. Moreover, σ_k is an and-node with $\sigma_k.status = \texttt{sat}$, hence $w.status \neq \texttt{unsat}$, meaning that $w.status = \texttt{sat}$. Setting $cn := u$ and setting σ to be a saturation path of w in G maintains the invariant.

By Remark 2, the branch formed by the instances of *cn* is an open branch of T.

Theorem 2. *Let $\mathcal{R}$ be a $\mathcal{SHI}$ RBox, $\mathcal{T}$ a TBox, and X a finite set of concepts. Let $G = \langle V, E \rangle$ be the graph constructed by Algorithm 2 for $(\mathcal{R}, \mathcal{T}, X)$, with $\tau \in V$ as the initial node. Then X is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$ iff $\tau.status = \texttt{sat}$.*

Proof. By Lemmas 6 and 7, X is consistent w.r.t. $(\mathcal{R}, \mathcal{T})$ iff $\tau.status = \texttt{sat}$ since $\tau.content = X \cup \mathcal{T}$. Since the calculus $C\mathcal{SHI}$ is sound and complete, it follows that X is satisfiable w.r.t. $(\mathcal{R}, \mathcal{T})$ iff $\tau.status = \texttt{sat}$.

Corollary 1. *Algorithm 2 is an EXPTIME decision procedure for $\mathcal{SHI}$.*

Proof. The EXPTIME complexity is established by Proposition 1.

6 Further Work and Conclusions

To the best of our knowledge, we have given the first EXPTIME tableau-based decision procedure for $\mathcal{SHI}$. The two essential features which allow our decision

procedure to have EXPTIME complexity are the analytic "future guessing" cut rules in combination with global caching.

Cut rules are usually considered expensive in tableau calculi because of their exponential nature and because their "blind guessing" produces nondeterminism. Although the "blind guessing" aspect is still prevalent in our cut rules, global caching means that an application of a cut rule in our and-or graph creates only an or-node, deterministically in polynomial time in the size of the graph, but does *not* require us to make two copies of the current and-or graph. Nor does it require us to later "determinise" the effects of "look behind with cut" as queried in the introduction. Indeed, by building a "use-check" into our procedure, and exploring the left branch of a cut rule first, we can avoid exploration of the right branch if the cut formula is not used in closing the left branch. Whether our method can be turned into practical EXPTIME decision procedures requires further investigation, but it obeys the basic principle that practical algorithms should be easy to implement and optimise. Besides, most known optimisation techniques for DL decision procedures are applicable since they are already incorporated into our decision procedure for $\mathcal{ALC}$ in [6], and it is easy to show that they transfer to our decision procedure for $\mathcal{SHI}$.

Soundness of global caching for $\mathcal{SHI}$ was not previously proved and was really an open problem. The idea of using analytic cut rules to reduce the complexity from NEXPTIME to EXPTIME was introduced in [2], but the "future guessing" form of our analytic cut rules is important and fundamentally different from the "look behind analytic cut" of [2].

Observe that the transformation of our $\mathcal{CSHI}$ calculus into an EXPTIME decision procedure for $\mathcal{SHI}$ is highly independent of the details of the calculus. The proof of the correctness of the transformation is also highly modular w.r.t. the calculus. We intend to formulate both at a more abstract level to apply our method for global caching to other modal and description logics.

References

1. Baader, F., Sattler, U.: An overview of tableau algorithms for description logics. Studia Logica 69, 5–40 (2001)
2. De Giacomo, G., Massacci, F.: Combining deduction and model checking into tableaux and algorithms for Converse-PDL. Information and Computation, pp. 117–137 pp. 87–138 (2000)
3. Ding, Y., Haarslev, V.: Tableau caching for description logics with inverse and transitive roles. In: Proc. DL-2006: International Workshop on Description Logics, pp. 143–149 (2006)
4. Donini, F., Massacci, F.: EXPTIME tableaux for ALC. Artificial Intelligence 124, 87–138 (2000)
5. Goré, R.: Tableau methods for modal and temporal logics. In: Agostino, D. (ed.) Handbook of Tableau Methods, pp. 297–396. Kluwer, Dordrecht (1999)
6. Goré, R., Nguyen, L.A.: Optimised EXPTIME tableaux for ALC using sound global caching, propagation and cutoffs. Manuscript (2007)

7. Heuerding, A., Seyfried, M., Zimmermann, H.: Efficient loop-check for backward proof search in some non-classical logics. In: Miglioli, P., Moscato, U., Ornaghi, M., Mundici, D. (eds.) TABLEAUX 1996. LNCS, vol. 1071, pp. 210–225. Springer, Heidelberg (1996)
8. Horrocks, I., Patel-Schneider, P.F.: Optimizing description logic subsumption. Journal of Logic and Computation 9(3), 267–293 (1999)
9. Horrocks, I., Sattler, U.: A description logic with transitive and inverse roles and role hierarchies. J. Log. Comput. 9(3), 385–410 (1999)
10. Nguyen, L.A.: Analytic tableau systems and interpolation for the modal logics KB, KDB, K5, KD5. Studia Logica 69(1), 41–57 (2001)
11. Rautenberg, W.: Modal tableau calculi and interpolation. JPL 12, 403–423 (1983)