ExpTime Tableaux with Global Caching for the Description Logic SHOQ

Linh Anh Nguyen[1,2] and Joanna Golińska-Pilarek[3]

[1] Institute of Informatics, University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
nguyen@mimuw.edu.pl
[2] Faculty of Information Technology, VNU University of Engineering and Technology
144 Xuan Thuy, Hanoi, Vietnam
[3] Institute of Philosophy, University of Warsaw
Krakowskie Przedmieście 3, 00-927 Warsaw, Poland
j.golinska@uw.edu.pl

July, 2013 (last revised: July, 2014)

Abstract. We give the first ExpTime (complexity-optimal) tableau decision procedure for checking satisfiability of a knowledge base in the description logic $\mathcal{SHOQ}$, which extends the basic description logic $\mathcal{ALC}$ with transitive roles, hierarchies of roles, nominals and quantified number restrictions. The complexity is measured using unary representation for numbers. Our procedure is based on global caching and integer linear feasibility checking.

Keywords: automated reasoning, description logics, global state caching, integer linear feasibility

1 Introduction

Description logics (DLs) are formal languages suitable for representing terminological knowledge. They are of particular importance in providing a logical formalism for ontologies and the Semantic Web. DLs represent the domain of interest in terms of concepts, individuals, and roles. A concept is interpreted as a set of individuals, while a role is interpreted as a binary relation among individuals. A knowledge base in a DL consists of axioms about roles (grouped into an RBox), terminology axioms (grouped into a TBox), and assertions about individuals (grouped into an ABox). A DL is usually specified by: i) a set of constructors that allow building complex concepts and complex roles from concept names, role names and individual names, ii) allowed forms of axioms and assertions. The basic DL $\mathcal{ALC}$ allows basic concept constructors listed in Table 1, but does not allow role constructors nor role axioms. The most common additional features for extending $\mathcal{ALC}$ are also listed in Table 1 together with syntax and examples: $\mathcal{I}$ is a role constructor, $\mathcal{Q}$ and $\mathcal{O}$ are concept constructors, while $\mathcal{H}$ and $\mathcal{S}$ are allowed forms of role axioms. The name of a DL is usually formed by the names of its additional features, as in the cases of $\mathcal{SH}$, $\mathcal{SHI}$, $\mathcal{SHIQ}$, $\mathcal{SHIO}$, $\mathcal{SHOQ}$ and $\mathcal{SHOIQ}$. $\mathcal{SROIQ}$ [12] is a further expressive DL used as the logical base for the Web Ontology Language OWL 2 DL.

Automated reasoning in DLs is useful, for example, in engineering and querying ontologies. One of basic reasoning problems in DLs is to check satisfiability of a knowledge base in a considered DL. Most of other reasoning problems in DLs are reducible to this one. In this paper, we study the problem of checking satisfiability of a knowledge base in the DL $\mathcal{SHOQ}$, which extends the basic DL $\mathcal{ALC}$ with transitive roles ($\mathcal{S}$), hierarchies of roles ($\mathcal{H}$), nominals ($\mathcal{O}$) and quantified number restrictions ($\mathcal{Q}$). It is known that this problem in $\mathcal{SHOQ}$ is ExpTime-complete [30] (even when numbers are coded in binary). Nominals, interpreted as singleton sets, are a useful notion to express identity and uniqueness. However, when interacting with inverse roles ($\mathcal{I}$) and quantified number restrictions in the DL $\mathcal{SHOIQ}$, they cause the complexity of the above mentioned problem to jump up to NExpTime-complete [29] (while that problem in any of the DLs $\mathcal{SHOQ}$, $\mathcal{SHIO}$, $\mathcal{SHIQ}$ is ExpTime-complete [30, 11, 29]).

In [13] Horrocks and Sattler gave a tableau algorithm for deciding the DL $\mathcal{SHOQ}$(D), which is the extension of $\mathcal{SHOQ}$ with concrete datatypes. Later, Pan and Horrocks [26] extended

Concept constructors of $\mathcal{ALC}$		
Constructor	Syntax	Example
complement	$\neg C$	$\neg Male$
intersection	$C \sqcap D$	$Human \sqcap Male$
union	$C \sqcup D$	$Doctor \sqcup Lawyer$
existential restriction	$\exists r.C$	$\exists hasChild.Male$
universal restriction	$\forall r.C$	$\forall hasChild.Female$
Additional constructors/features of other DLs		
Constructor/Feature	Syntax	Example
inverse roles ($\mathcal{I}$)	r^-	$hasChild^-$ (i.e., $hasParent$)
quantified number restrictions ($\mathcal{Q}$)	$\geq n\, R.C$ $\leq n\, R.C$	$\geq 3\, hasChild.Male$ $\leq 2\, hasParent.\top$
nominals ($\mathcal{O}$)	$\{a\}$	$\{John\}$
hierarchies of roles ($\mathcal{H}$)	$R \sqsubseteq S$	$hasChild \sqsubseteq hasDescendant$
transitive roles ($\mathcal{S}$)	$R \circ R \sqsubseteq R$	$hasDescendant \circ hasDescendant \sqsubseteq hasDescendant$

Table 1. Concept constructors for $\mathcal{ALC}$ and some additional constructors/features of other DLs.

the method of [13] to give a tableau algorithm for deciding the DL $\mathcal{SHOQ}(\mathrm{D}_n)$, which is the extension of $\mathcal{SHOQ}$ with n-ary datatype predicates. These algorithms use backtracking to deal with disjunction ($\sqcup$) and "or"-branching (e.g., the "choice"-rule) and use a straightforward way for dealing with quantified number restrictions. They have a non-optimal complexity (N2ExpTime) when numbers are coded in unary.[4] In [2] Faddoul and Haarslev gave an algebraic tableau reasoning algorithm for $\mathcal{SHOQ}$, which combines the tableau method with linear integer programming. The aim was to increase efficiency of handling quantified number restrictions. However, their algorithm still uses backtracking to deal with disjunction and "or"-branching and has a non-optimal complexity ("double exponential" [2]).

This paper is a revised and extended version of our workshop paper [20]. In this work we present the *first* tableau method with an ExpTime (optimal) complexity for checking satisfiability of a knowledge base in the DL $\mathcal{SHOQ}$ when numbers are coded in unary.[5] Our method is based on global caching and integer linear feasibility checking.

The idea of global caching comes from Pratt's work [27] on PDL. It was formally formulated for tableaux in some DLs in [6, 8] and has been applied to several modal and description logics [5, 7, 21–25, 1] to obtain tableau decision procedures with an optimal complexity. A variant of global caching, called global state caching, was used to obtain cut-free optimal tableau decision procedures for several modal logics with converse and DLs with inverse roles [9, 10, 16, 17, 19].

Integer linear programming was exploited for tableaux in [3, 2] to increase efficiency of reasoning with quantified number restrictions. However, the first work that applied integer linear feasibility checking to tableaux was [18, 19]. In [18], Nguyen gave the first ExpTime (optimal) tableau decision procedure for checking satisfiability of a knowledge base in the DL $\mathcal{SHIQ}$ when numbers are coded in unary. His procedure is based on global state caching and integer linear feasibility checking. In the current paper, we apply his method of integer linear feasibility checking to $\mathcal{SHOQ}$. The adaptation requires special techniques due to the following reasons: i) we use global caching for $\mathcal{SHOQ}$, while Nguyen's work [18] uses global state caching for $\mathcal{SHIQ}$ (for dealing with inverse roles); ii) we have to deal with the interaction between number restrictions and nominals. Our method substantially differs from Farsiniamarj's method of exploiting integer programming for tableaux [3]. Our technique for dealing with both nominals and quantified number restrictions is also essentially different from the one by Faddoul and Haarslev [2].

The rest of this paper is structured as follows. In Section 2 we recall notation and semantics of $\mathcal{SHOQ}$ as well as the integer feasibility problem for DLs [18]. In Section 4 we present our tableau

[4] When the algorithms are improved by using "anywhere blocking", the complexity will be NExpTime and still non-optimal.

[5] This corrects the claim of [20] that the complexity is measured using binary representation for numbers.

decision procedure for $\mathcal{SHOQ}$ together with examples for illustrating our tableau method. We conclude this work in Section 5. Proofs for our results are given in the Appendix.

2 Preliminaries

2.1 Notation and Semantics of $\mathcal{SHOQ}$

Our language uses a finite set $\mathbf{C}$ of *concept names*, a finite set $\mathbf{R}$ of *role names*, and a finite set $\mathbf{I}$ of *individual names*. We use letters like A and B for concept names, r and s for role names, and a and b for individual names. We also refer to A and B as *atomic concepts*, to r and s as *roles*, and to a and b as *(named) individuals*.

An ($\mathcal{SHOQ}$) *RBox* $\mathcal{R}$ is a finite set of role axioms of the form $r \sqsubseteq s$ or $r \circ r \sqsubseteq r$. For example, $\mathit{link} \sqsubseteq \mathit{path}$ and $\mathit{path} \circ \mathit{path} \sqsubseteq \mathit{path}$ are such role axioms.

By $\mathit{ext}(\mathcal{R})$ we denote the least extension of $\mathcal{R}$ such that:

- $r \sqsubseteq r \in \mathit{ext}(\mathcal{R})$ for any role r
- if $r \sqsubseteq r' \in \mathit{ext}(\mathcal{R})$ and $r' \sqsubseteq r'' \in \mathit{ext}(\mathcal{R})$ then $r \sqsubseteq r'' \in \mathit{ext}(\mathcal{R})$.

We write $r \sqsubseteq_{\mathcal{R}} s$ to denote $r \sqsubseteq s \in \mathit{ext}(\mathcal{R})$, and $\mathit{trans}_{\mathcal{R}}(r)$ to denote $(r \circ r \sqsubseteq r) \in \mathit{ext}(\mathcal{R})$. If $r \sqsubseteq_{\mathcal{R}} s$ then r is a *subrole* of s (w.r.t. $\mathcal{R}$). If $\mathit{trans}_{\mathcal{R}}(s)$ then s is a *transitive role* (w.r.t. $\mathcal{R}$). A role is *simple* (w.r.t. $\mathcal{R}$) if it is neither transitive nor has any transitive subrole (w.r.t. $\mathcal{R}$).

Concepts in $\mathcal{SHOQ}$ are formed using the following BNF grammar, where n is a nonnegative integer and s is a simple role:

$$C, D ::= \top \mid \bot \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists r.C \mid \forall r.C \mid \{a\} \mid \geq n\, s.C \mid \leq n\, s.C$$

A concept stands for a set of individuals. The concept $\top$ stands for the set of all individuals (in the considered domain). The concept $\bot$ stands for the empty set. The constructors $\neg$, $\sqcap$ and $\sqcup$ stand for the set operators: complement, intersection and union. For the remaining forms, we just give some illustrative examples: $\exists \mathit{hasChild}.\mathit{Male}$, $\forall \mathit{hasChild}.\mathit{Female}$, $\geq 2\, \mathit{hasChild}.\mathit{Teacher}$, $\leq 5\, \mathit{hasChild}.\top$.

We use letters like C and D to denote arbitrary concepts.

A *TBox* is a finite set of axioms of the form $C \sqsubseteq D$ or $C \doteq D$.

An *ABox* is a finite set of assertions of the form $a\!:\!C$, $r(a, b)$ or $a \neq b$. An *eABox* (*extended ABox*) is a finite set of assertions of the form $a\!:\!C$, $r(a, b)$, $\neg r(a, b)$, $a \doteq b$ or $a \neq b$.

An axiom $C \sqsubseteq D$ means C is a subconcept of D, while $C \doteq D$ means C and D are equivalent concepts. An assertion $a\!:\!C$ means a is an instance of concept C, $r(a, b)$ means the pair $\langle a, b\rangle$ is an instance of role r, and $a \neq b$ means a and b are distinct individuals.

A *knowledge base* in $\mathcal{SHOQ}$ is a tuple $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$, where $\mathcal{R}$ is an RBox, $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox.

We say that a role s is *numeric* w.r.t. a knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ if:

- it is simple w.r.t. $\mathcal{R}$ and occurs in a concept $\geq n\, s.C$ or $\leq n\, s.C$ in KB, or
- $s \sqsubseteq_{\mathcal{R}} r$ and r is numeric w.r.t. KB.

We will simply call such an s a numeric role when KB is clear from the context.

A *formula* is defined to be either a concept or an eABox assertion. We use letters like φ, ψ and ξ to denote formulas. Let $\mathsf{null}\!:\!C$ stand for C. We use α to denote either an individual or null. Thus, $\alpha\!:\!C$ is a formula of the form $a\!:\!C$ or $\mathsf{null}\!:\!C$ (which means C).

An *interpretation* $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$ consists of a non-empty set $\Delta^{\mathcal{I}}$, called the *domain* of $\mathcal{I}$, and a function $\cdot^{\mathcal{I}}$, called the *interpretation function* of $\mathcal{I}$, that maps each concept name A to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name r to a binary relation $r^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$, and each individual name a to an

element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The interpretation function is extended to complex concepts as follows, where $\sharp Z$ denotes the cardinality of a set Z:

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \qquad \bot^{\mathcal{I}} = \emptyset \qquad (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} - C^{\mathcal{I}}$$
$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \qquad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \qquad \{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$$
$$(\exists r.C)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \exists y\big[\langle x, y\rangle \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\big]\big\}$$
$$(\forall r.C)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \forall y\big[\langle x, y\rangle \in r^{\mathcal{I}} \text{ implies } y \in C^{\mathcal{I}}\big]\big\}$$
$$(\geq n\, s.C)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid \langle x, y\rangle \in s^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \geq n\big\}$$
$$(\leq n\, s.C)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid \langle x, y\rangle \in s^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \leq n\big\}.$$

For a set Γ of concepts, define $\Gamma^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid x \in C^{\mathcal{I}} \text{ for all } C \in \Gamma\}$.

The relational composition of binary relations R_1 and R_2 is denoted by $R_1 \circ R_2$.

An interpretation $\mathcal{I}$ is a *model of an RBox* $\mathcal{R}$ if for every axiom $r \sqsubseteq s$ (resp. $r \circ r \sqsubseteq r$) of $\mathcal{R}$, we have that $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$ (resp. $r^{\mathcal{I}} \circ r^{\mathcal{I}} \subseteq r^{\mathcal{I}}$). Note that if $\mathcal{I}$ is a model of $\mathcal{R}$ then it is also a model of $ext(\mathcal{R})$.

An interpretation $\mathcal{I}$ is a *model of a TBox* $\mathcal{T}$ if for every axiom $C \sqsubseteq D$ (resp. $C \doteq D$) of $\mathcal{T}$, we have that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ (resp. $C^{\mathcal{I}} = D^{\mathcal{I}}$).

Given an interpretation $\mathcal{I}$, define:

$$\begin{array}{ll} \mathcal{I} \models a{:}C & \text{iff } a^{\mathcal{I}} \in C^{\mathcal{I}} \\ \mathcal{I} \models r(a,b) & \text{iff } \langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in r^{\mathcal{I}} \\ \mathcal{I} \models \neg r(a,b) & \text{iff } \langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \notin r^{\mathcal{I}} \\ \mathcal{I} \models a \doteq b & \text{iff } a^{\mathcal{I}} = b^{\mathcal{I}} \\ \mathcal{I} \models a \neq b & \text{iff } a^{\mathcal{I}} \neq b^{\mathcal{I}}. \end{array}$$

If $\mathcal{I} \models \varphi$ then we say that $\mathcal{I}$ *satisfies* φ. An interpretation $\mathcal{I}$ is a *model of an eABox* $\mathcal{A}$ if it satisfies all the assertions of $\mathcal{A}$. In that case, we also say that $\mathcal{I}$ *satisfies* $\mathcal{A}$.

An interpretation $\mathcal{I}$ is a *model of a knowledge base* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ if $\mathcal{I}$ is a model of $\mathcal{R}$, $\mathcal{T}$ and $\mathcal{A}$. A knowledge base $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ is *satisfiable* if it has a model.

An interpretation $\mathcal{I}$ *satisfies* a concept C (resp. a set X of concepts) if $C^{\mathcal{I}} \neq \emptyset$ (resp. $X^{\mathcal{I}} \neq \emptyset$). It *validates* a concept C if $C^{\mathcal{I}} = \Delta^{\mathcal{I}}$. A set X of concepts is *satisfiable w.r.t. an RBox* $\mathcal{R}$ *and a TBox* $\mathcal{T}$ if there exists a model of $\mathcal{R}$ and $\mathcal{T}$ that satisfies X. We say that an eABox $\mathcal{A}$ is *satisfiable w.r.t. an RBox* $\mathcal{R}$ *and a TBox* $\mathcal{T}$ if there exists an interpretation $\mathcal{I}$ that is a model of $\mathcal{A}$, $\mathcal{R}$ and $\mathcal{T}$. In that case, we also call $\mathcal{I}$ a model of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$.

In this paper, we assume that concepts and ABox assertions are represented in negation normal form (NNF), where $\neg$ occurs only directly before atomic concepts.[6] We use $\overline{C}$ to denote the NNF of $\neg C$, and for $\varphi = (a{:}C)$, we use $\overline{\varphi}$ to denote $a{:}\overline{C}$. For simplicity, we treat axioms of a TBox $\mathcal{T}$ as concepts representing global assumptions: an axiom $C \sqsubseteq D$ is treated as $\overline{C} \sqcup D$, while an axiom $C \doteq D$ is treated as $(\overline{C} \sqcup D) \sqcap (\overline{D} \sqcup C)$.[7] That is, we assume that $\mathcal{T}$ consists of concepts in NNF. A concept $C \in \mathcal{T}$ can be thought of as an axiom $\top \sqsubseteq C$. Thus, an interpretation $\mathcal{I}$ is a model of $\mathcal{T}$ iff $\mathcal{I}$ validates every concept $C \in \mathcal{T}$.

2.2 An Integer Feasibility Problem for Description Logics

For dealing with number restrictions in $\mathcal{SHOQ}$, we consider the following integer feasibility problem, which was introduced in [18]:

$$\sum_{j=1}^{m} a_{i,j} \cdot x_j \bowtie_i b_i, \quad \text{for } 1 \leq i \leq l;$$
$$x_j \geq 0, \quad \text{for } 1 \leq j \leq m;$$

[6] Every formula can be transformed to an equivalent formula in NNF in polynomial time.

[7] As this way of handling the TBox is not efficient in practice, the absorption technique like the one discussed in [17] can be used to improve the performance of reasoning.

where each $a_{i,j}$ is either 0 or 1, each x_j is a variable standing for a natural number, each $\bowtie_i$ is either $\leq$ or $\geq$, each b_i is a natural number encoded by using no more than n bits (i.e., $b_i \leq 2^n$). We call this an IFDL(l, m, n)-problem (a problem of Linear Integer Feasibility for Description Logics with size specified by l, m, n). The problem is *feasible* if it has a solution (i.e., values for the variables x_j, $1 \leq j \leq l$, that are natural numbers satisfying the constraints), and is *infeasible* otherwise. By solving an IFDL(l, m, n)-problem we mean checking its feasibility.

It is known from linear programming that, if the variables x_j are not required to be natural numbers but can be real numbers then the above feasibility problem can be solved in polynomial time in l, m and n. The general integer linear optimization problem is known to be NP-hard.[8]

To solve an integer feasibility problem, we propose to use the decomposition technique and the "branch and bound" method [14]. One can first analyze dependencies between the variables and the constraints to decompose the problem into smaller independent subproblems, then solve the subproblems that are trivial, and after that apply the "branch and bound" method [14] to the remaining subproblems.

The above mentioned approach may not guarantee that a given IFDL(l, m, n)-problem is solved in exponential time in n. We recall below an estimation of the upper bound for the complexity for some specific cases, using another approach.

Lemma 2.1 ([18]). *Every IFDL(l, m, n)-problem such that $l \leq n$, m is (at most) exponential in n, and $b_i \leq n$ for all $1 \leq i \leq l$ can be solved in (at most) exponential time in n.*

Proof. Consider the following nondeterministic procedure:

1. initialize $c_{i,j} := 0$ for each $1 \leq i \leq l$ and $1 \leq j \leq m$ such that $a_{i,j} = 1$
2. for each i from 1 to l do
   for each k from 1 to b_i do
   choose some j among $1, \ldots, m$ such that $a_{i,j} = 1$ and set $c_{i,j} := c_{i,j} + 1$
3. if the set of constraints $\{x_j \bowtie_i c_{i,j} \mid 1 \leq i \leq l, 1 \leq j \leq m, a_{i,j} = 1\}$ is feasible then return "yes", else return "no".

Observe that the considered IFDL(l, m, n)-problem is feasible iff there exists a run of the above procedure that returns "yes". Since $b_i \leq n$ for all $1 \leq i \leq l$, there are no more than $m^{l \cdot n}$ possible runs of the above procedure. All the steps of the procedure can be executed in time $O(l \cdot m \cdot n)$. Since $l \leq n$ and m is (at most) exponential in n, we conclude that the considered IFDL(l, m, n)-problem can deterministically be solved in (at most) exponential time in n. □

The following lemma is more general than the above lemma.

Lemma 2.2 ([18]). *Every IFDL(l, m, n)-problem satisfying the following properties can be solved in (at most) exponential time in n:*

- *$l \leq n$, m is (at most) exponential in n,*
- *and*
  - *either $b_i \leq n$ for all $1 \leq i \leq l$ such that $\bowtie_i$ is $\leq$*
  - *or $b_i \leq n$ for all $1 \leq i \leq l$ such that $\bowtie_i$ is $\geq$.*

Proof. Suppose $l \leq n$, m is (at most) exponential in n, and $b_i \leq n$ for all $1 \leq i \leq l$ such that $\bowtie_i$ is $\leq$. The other case is similar and omitted. Consider the following nondeterministic procedure:

1. let $J = \{j \mid 1 \leq j \leq m$ and there exists $1 \leq i \leq l$ such that $\bowtie_i$ is $\leq$ and $a_{i,j} = 1\}$
2. for each $1 \leq i \leq l$ and $1 \leq j \leq m$ such that $\bowtie_i$ is $\leq$ and $a_{i,j} = 1$, set $c_{i,j} := 0$
3. for each i from 1 to l such that $\bowtie_i$ is $\leq$, do
   for each k from 1 to b_i do
   choose some j among $1, \ldots, m$ such that $a_{i,j} = 1$ and set $c_{i,j} := c_{i,j} + 1$

[8] `http://en.wikipedia.org/wiki/Integer_programming`

4. for each $j \in J$ do
   $d_j := \min\{c_{i,j} \mid 1 \leq i \leq l,\ \bowtie_i \text{ is } \leq \text{ and } a_{i,j} = 1\}$
5. if the set of constraints $\{\sum_{j=1}^{m} a_{i,j} \cdot x_j \geq b_i \mid 1 \leq i \leq l,\ \bowtie_i \text{ is } \geq\} \cup \{x_j = d_j \mid j \in J\}$ is feasible then return "yes", else return "no".

Observe that the considered IFDL(l, m, n)-problem is feasible iff there exists a run of the above procedure that returns "yes". Under the assumptions of the lemma, there are no more than $m^{l \cdot n}$ possible runs of the above procedure. All the steps of the procedure can be executed in time $O(l \cdot m \cdot n)$. Since $l \leq n$ and m is (at most) exponential in n, we conclude that the considered IFDL(l, m, n)-problem can deterministically be solved in (at most) exponential time in n. □

3 The Traditional Tableau Method and Its Problems

The problem we study is to check whether a given knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ in $\mathcal{SHOQ}$ is satisfiable. The *traditional tableau method* for this task is as follows. We start from the ABox $\mathcal{A}$ and try to modify it to obtain a model of KB. At each moment, we have an ABox, which is like a graph. At the beginning, each (named) individual occurring in $\mathcal{A}$ is a node labeled by the set $\mathit{Label}(a) = \{C \mid a{:}C \in \mathcal{A}\} \cup \mathcal{T}$, and each assertion $r(a, b)$ in $\mathcal{A}$ forms an edge from a to b that is labeled by r. The concepts in $\mathit{Label}(a)$ are treated as requirements to be realized for a. As $\mathcal{T}$ consists of the global assumptions that should be satisfied for all individuals, the concepts from $\mathcal{T}$ are included in $\mathit{Label}(a)$. For example, an axiom $\top \sqsubseteq \mathit{Human}$ is encoded in NNF as Human, and such a global assumption states that all individuals in the domain should be human beings. To see how the requirements for nodes can be realized, let us consider several cases:

- If $C \sqcap D \in \mathit{Label}(v)$ then to realize the requirement $C \sqcap D$ for v we add both C and D to $\mathit{Label}(v)$. To see the intuition of this, assume that *John* is an individual and $\mathit{Male} \sqcap \mathit{Happy} \in \mathit{Label}(\mathit{John})$. In this case, *John* is required to satisfy the property $\mathit{Male} \sqcap \mathit{Happy}$, and to realize this, we add both the requirements *Male* and *Happy* to $\mathit{Label}(\mathit{John})$.
- If $C \sqcup D \in \mathit{Label}(v)$ then to realize the requirement $C \sqcup D$ for v we add either C or D to $\mathit{Label}(v)$. That is, we make an "or"-branching, which is dealt with by backtracking (since at each moment we consider only one ABox). If the current "or"-branch leads to inconsistency, we will backtrack to the nearest "or"-branching point and try another "or"-branch. To see the intuition of this, assume that $\mathit{Doctor} \sqcup \mathit{Lawyer} \in \mathit{Label}(\mathit{John})$. In this case, *John* is required to satisfy the property $\mathit{Doctor} \sqcup \mathit{Lawyer}$, which states that he is either a doctor or a lawyer, and to realize this requirement, we make a choice: either add the requirement *Doctor* or add the requirement *Lawyer* to $\mathit{Label}(\mathit{John})$.
- If $\exists r.C \in \mathit{Label}(v)$ then to realize the requirement $\exists r.C$ for v we connect v to a new node w with $\mathit{Label}(w) = \{C\} \cup \mathcal{T}$ via an edge labeled by r. Once again, $\mathcal{T}$ is included in $\mathit{Label}(w)$ because it consists of the global assumptions that should be realized for all individuals. (Instead of creating a new node, one may use an existing node for w as in the approach with global caching, but this should be done appropriately, e.g., as in our tableau method discussed in the next section. Alternatively, one can use a blocking technique as in [13, 26].) To see the intuition of the expansion, assume that $\exists \mathit{hasChild}.\mathit{Female} \in \mathit{Label}(\mathit{John})$. In this case, *John* should satisfy the requirement $\exists \mathit{hasChild}.\mathit{Female}$, which states that he has a female child (a daughter). To realize this, we connect the node *John* of the graph to a new node w with $\mathit{Label}(w) = \{\mathit{Female}\} \cup \mathcal{T}$ via an edge labeled by *hasChild*. From this, it can be seen that the graph contains not only named individuals occurring in $\mathcal{A}$, but it may also contain nodes like w, which are called unnamed individuals.
- If $\forall r.C \in \mathit{Label}(v)$ then to realize the requirement $\forall r.C$ for v, for every node w such that there is an edge with the label r from v to w, we add C to $\mathit{Label}(w)$. To see the intuition of this, assume that $\forall \mathit{hasChild}.\mathit{Happy} \in \mathit{Label}(\mathit{John})$ and there are edges with the label

hasChild from the node *John* to the nodes *Mary* and w (i.e., *Mary* and w are children of *John*). In this case, *John* should satisfy the requirement $\forall \mathit{hasChild}.\mathit{Happy}$, which states that all the children of *John* should be happy. To realize this, we add the requirement *Happy* to both $\mathit{Label}(\mathit{Mary})$ and $\mathit{Label}(w)$.

- If $\{a\} \in \mathit{Label}(v)$ then v and a should denote the same individual (this is the semantics of nominals), and to realize the requirement $\{a\}$ for v we merge the nodes v and a together in an appropriate way.
- If $\geq n\, r.C \in \mathit{Label}(v)$ then to realize the requirement $\geq n\, r.C$ for v we connect v to n new nodes $w_1, \ldots, w_n$ with $\mathit{Label}(w_i) = \{C\} \cup \mathcal{T}$ via an edge labeled by r for all $1 \leq i \leq n$, and keep the constraints $w_i \neq\!\!\!\!\cdot\; w_j$ for all $1 \leq i \neq j \leq n$. (Once again, an appropriate caching or blocking technique can be used to reduce the number of created nodes.) To see the intuition of the expansion, assume that $\geq 2\, \mathit{hasChild}.\mathit{Female} \in \mathit{Label}(\mathit{John})$. In this case, *John* should satisfy the requirement $\geq 2\, \mathit{hasChild}.\mathit{Female}$, which states that he has at least two female children (daughters). To realize this, we connect the node *John* of the graph to new nodes w_1 and w_2 with $\mathit{Label}(w_1) = \mathit{Label}(w_2) = \{\mathit{Female}\} \cup \mathcal{T}$ via an edges labeled by *hasChild* and keep the constraint $w_1 \neq\!\!\!\!\cdot\; w_2$.
- If $\leq n\, r.C \in \mathit{Label}(v)$ and there are pairwise different nodes $w_1, \ldots, w_{n+1}$ such that v is connected to w_i via an edge labeled by r and $C \in \mathit{Label}(w_i)$ for all $1 \leq i \leq n+1$, then:
  - if there exist different i and j among $1, \ldots, n$ such that the constraint $w_i \neq\!\!\!\!\cdot\; w_j$ is absent then we merge w_i and w_j together in an appropriate way,
  - otherwise, the current ABox is inconsistent and we do backtracking.

Inconsistency may occur, for example, in the following cases:

- when $\bot \in \mathit{Label}(v)$ for some v; or
- when $\{A, \neg A\} \subseteq \mathit{Label}(v)$ for some A and v; or
- when a and b were merged together, but the assertion $a \neq\!\!\!\!\cdot\; b$ is a kept constraint; or
- when the current ABox contains an edge with the label r from a to b, but $(\neg r(a,b)) \in \mathcal{A}$.

As mentioned before, when the current ABox is inconsistent, backtracking occurs, and if there is no "or"-branching point to come back, the process terminates with the result "*KB* is unsatisfiable".

The above discussion only gives a sketch on how the traditional tableau method works. We did not discuss how role axioms can be dealt with and how a blocking technique can be applied to guarantee termination. Furthermore, merging nodes causes merging edges, and hence an edge may be labeled by a set of roles. In general, a tableau algorithm is usually designed so that, if it does not terminate with the result "*KB* is unsatisfiable", then *KB* is satisfiable and we can directly construct a model of *KB* from the resulting (*clash-free* and *completed*) ABox. We refer the reader to [13, 26] for details.

The traditional tableau method for $\mathcal{SHOQ}$ has the advantage of being intuitive, but it has two disadvantages that make the complexity non-optimal (N2ExpTime or NExpTime, depending on the applied blocking technique, in comparison with the optimal complexity ExpTime) and the reasoning process not scalable w.r.t. number restrictions:

- An ABox is like an "and"-structure (i.e., all of its assertions must hold together) and the search space for the traditional tableau method is an "or"-tree of "and"-structures. Recall that "or"-branchings are caused, amongst others, by the rule for realizing requirements of the form $C \sqcup D$. The problem is that two nodes in ABoxes in different "or"-branches may have the same label and the same "neighborhood", and both of them are expanded with no reuse, which causes a kind of redundant computation [8].
- Reconsider the traditional tableau rule for realizing a requirement of the form $\geq n\, r.C$. If n is big, for example, 1000 or 1000000, then the rule creates many new nodes. In the DL literature, this is called "pay-as-you-go", but this payment is unnecessarily too high when n is big and it causes the reasoning process not scalable w.r.t. number restrictions.

4 ExpTime Tableaux for $\mathcal{SHOQ}$

In this section, we first define the data structures and outline the framework of our tableau method. We then describe our techniques for dealing with nominals. After that, we specify the used tableau rules and state properties of the resulting tableau decision procedure.

4.1 Data Structures and the Tableau Framework

Recall that the search space for the traditional tableau method for $\mathcal{SHOQ}$ [13, 26] is an "or"-tree of "and"-structures, and this causes the complexity of the reasoning process to become non-optimal (even in the case without number restrictions). The idea for overcoming this problem is to use global caching [27, 8, 17]. With global caching, the search space is like a single "and-or" graph. For checking satisfiability of a concept w.r.t. an RBox and a TBox [8], each node of the graph is a *simple node* like an individual (in an ABox). For checking satisfiability of a knowledge base [17, 18], each node of the graph is either a **complex node** like an eABox, or a **simple node** like an individual. More precisely, the label of a complex node is a set of eABox assertions, while the label of a simple node is a set of concepts. The information about whether a node v is complex or simple is kept by $SType(v)$ (the **subtype of** v).

At the beginning, the graph has only one node, called the **root**, which is a complex node. Then, in the first stage, complex nodes are expanded only by so called **static (tableau) rules** that do not create new (unnamed) individuals. This creates a layer of complex nodes. When no static tableau rules are applicable to a complex node v, if $Label(v)$ contains a requirement of the form $a : \exists r.C$ then to realize this requirement we can connect v to a *simple* node w with $Label(w) = \{C\} \cup \mathcal{T}$ via an edge e. This edge is related to a and r. To keep this information we store $\pi_I(e) = a$ (the letter π stands for "projection" and the letter I stands for "individual") and $\pi_R(e) = \{r\}$ (the letter R stands for "roles"; as mentioned earlier, due to merging nodes, an edge may be labeled by more than one role, and hence we use a set of roles).

A **transitional (tableau) rule** is a rule that realizes a requirement of the form $a : \exists r.C$, $a : (\geq n\, r.C)$, $\exists r.C$ or $\geq n\, r.C$ for a node v by connecting v to a new node or a number of new nodes or by using some existing nodes. If no static rules are applicable to a node v then v is called a **state**, otherwise it is called a **non-state**. This information is kept by $Type(v)$ (the **type of** v). Transitional tableau rules are applied only to states. A non-state is like an "or"-node in an "and-or" graph, but a state is a structure more sophisticated than an "and"-node in an "and-or" graph (due to feasibility checking of the set of integer linear constraints related to the state).[9]

Consider a simple state v (i.e., a state that is a simple node) with $\exists r.C \in Label(v)$. To realize this requirement for v, we can connect v to a new simple node w with $Label(w) = \{C\} \cup \mathcal{T}$ by an edge e. For such an edge e, let $\pi_I(e) = \mathsf{null}$ (i.e., no named individual is related to e).

Consider a state v. To realize requirements of the form $a : (\geq n\, r.C)$, $a : (\leq n\, r.C)$, $\geq n\, r.C$ or $\leq n\, r.C$ for v, we may have to connect v to some simple nodes w_i by edges e_i, respectively, and check feasibility of a certain set of integer linear constraints. The **set of integer linear constraints for** v is kept by $ILConstraints(v)$. For such mentioned edges e_i, let $\pi_T(e_i) = \mathsf{checkingFeasibility}$ (the letter T stands for "type"). For other edges e, which are created for realizing a requirement of the form $a : \exists r.C$ or $\exists r.C$, let $\pi_T(e) = \mathsf{testingClosedness}$.

We have explained the attributes $\pi_T(e)$, $\pi_R(e)$ and $\pi_I(e)$ that should be kept for an edge e outgoing from a state. Summing up, we have the following formal definition:

Definition 4.1. Let $EdgeLabels = \{\mathsf{testingClosedness}, \mathsf{checkingFeasibility}\} \times \mathcal{P}(\mathbf{R}) \times (\mathbf{I} \cup \{\mathsf{null}\})$. For $e \in EdgeLabels$, let $e = \langle \pi_T(e), \pi_R(e), \pi_I(e) \rangle$. Thus, $\pi_T(e)$ is called the type of the edge label e, $\pi_R(e)$ is a set of roles, and $\pi_I(e)$ is either an individual or null. (Each edge is specified by the source, the target and the label.) □

[9] In tableaux for simpler DLs like $\mathcal{ALC}$ [8] or $\mathcal{SHI}$ [17], a state is simply an "and"-node.

We have explained the attributes $Label(v)$, $Type(v)$, $SType(v)$ and $ILConstraints(v)$ for a node v. We need three more attributes for v, which are described and justified below.

- To realize the requirement $C \sqcup D \in Label(v)$ for a simple node v, we expand v by a static rule that connects v to two simple nodes w_1 and w_2 such that $Label(w_1) = Label(v) \cup \{C\} \setminus \{C \sqcup D\}$ and $Label(w_2) = Label(v) \cup \{D\} \setminus \{C \sqcup D\}$. The requirement $C \sqcup D$ is put to the sets $RFmls(w_1)$ and $RFmls(w_2)$ to record that it has been realized for w_1 and w_2, respectively. In general, the attribute $RFmls(w)$ for a node w keeps the set of the requirements that have been realized for w by using static rules. It is called the **set of reduced formulas of** w.
- Suppose v is a complex node and either $a \doteq b$ or $a{:}\{b\}$ belongs to $Label(v)$. Then, to realize that requirement for v, we merge the individual b to the individual a in an appropriate way and record this fact by keeping $IndRepl(v)(b) = a$. The attribute $IndRepl(v)$ is called the **partial mapping specifying replacements of individuals for the node** v.
- The last attribute needed for a node v is called the **status of** v and denoted by $Status(v)$. Possible statuses of nodes are: unexpanded, partially-expanded, fully-expanded, closed, open, blocked, and closed w.r.t. a set of complex states. Informally, closed means "unsatisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$", open means "satisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$", and closed-wrt(U) means "unsatisfiable w.r.t. $\mathcal{R}$, $\mathcal{T}$ and any node from U".

We arrive at the following formal definition.

Definition 4.2. A *tableau* is a rooted graph $G = \langle V, E, \nu \rangle$, where V is a set of nodes, $E \subseteq V \times V$ is a set of edges, $\nu \in V$ is the root, each node $v \in V$ has a number of attributes, and each edge $\langle v, w \rangle$ may have a number of labels from $EdgeLabels$.[10] The attributes of a tableau node v are:

- $Type(v) \in \{\mathsf{state}, \mathsf{non\text{-}state}\}$.
- $SType(v) \in \{\mathsf{complex}, \mathsf{simple}\}$ is called the subtype of v.
- $Status(v) \in \{\mathsf{unexpanded}, \mathsf{p\text{-}expanded}, \mathsf{f\text{-}expanded}, \mathsf{closed}, \mathsf{open}, \mathsf{blocked}\} \cup \{\mathsf{closed\text{-}wrt}(U) \mid U \subseteq V \text{ and } Type(u) = \mathsf{state} \wedge SType(u) = \mathsf{complex} \text{ for all } u \in U\}$, where p-expanded and f-expanded mean "partially expanded" and "fully expanded", respectively. $Status(v)$ may be p-expanded only when $Type(v) = \mathsf{state}$. If $Status(v) = \mathsf{closed\text{-}wrt}(U)$ then we say that the node v is closed w.r.t. any node from U.
- $Label(v)$ is a finite set of formulas called the label of v.
- $RFmls(v)$ is a finite set of formulas called the set of reduced formulas of v.
- $IndRepl(v) : \mathbf{I} \to \mathbf{I}$ is a partial mapping specifying replacements of individuals. It is available only when v is a complex node. If $IndRepl(v)(a) = b$ then at the node v we have $a \doteq b$ and b is the representative of its equivalence class.
- $ILConstraints(v)$ is a set of integer linear constraints. It is available only when $Type(v) = \mathsf{state}$. The constraints use variables $x_{w,e}$ indexed by a pair $\langle w, e \rangle$ such that $\langle v, w \rangle \in E$, $e \in ELabels(v, w)$ and $\pi_T(e) = \mathsf{checkingFeasibility}$. Such a variable specifies how many copies of the successor w using the edge label e will be created for v. □

If $\langle v, w \rangle \in E$ then we call v a *predecessor* of w and w a *successor* of v. An edge outgoing from a node v has labels iff $Type(v) = \mathsf{state}$. When defined, the set of labels of an edge $\langle v, w \rangle$ is denoted by $ELabels(v, w)$. If $e \in ELabels(v, w)$ then $\pi_I(e) = \mathsf{null}$ iff $SType(v) = \mathsf{simple}$.

Formally, a node v is called a *state* if $Type(v) = \mathsf{state}$, and a *non-state* otherwise. It is called a *complex node* if $SType(v) = \mathsf{complex}$, and a *simple node* otherwise. The root ν is a complex non-state.

A node may have status blocked only when it is a simple node with the label containing a nominal $\{a\}$. The status blocked can be updated only to closed or closed-wrt(...). We write closed-wrt(...) to mean closed-wrt(U) for some U. By $Status(v) \neq \mathsf{closed\text{-}wrt}(\{u, \ldots\})$ we denote that $Status(v)$ is not of the form closed-wrt(U) with $u \in U$.

[10] An edge $\langle v, w \rangle$ may have a number of labels from $EdgeLabels$ because of global caching, which we will briefly discuss later.

Function ConToSucc($v, type, sType, label, rFmls, indRepl, eLabel$)

Global data: a rooted graph $\langle V, E, \nu \rangle$.
Purpose: connect a node v to a successor, which is created if necessary.

if $v \neq$ null *and there exists* $w \in V$ *such that* $Label(w) = label$ *and* ($Type(w) = type$ *or* $SType(w) =$ simple) **then**
  $E := E \cup \{\langle v, w \rangle\}$, $RFmls(w) := RFmls(w) \cup rFmls$;
  if $Type(v) =$ state **then** $ELabels(v, w) := ELabels(v, w) \cup \{eLabel\}$;
else
  create a new node w, $V := V \cup \{w\}$, **if** $v \neq$ null **then** $E := E \cup \{\langle v, w \rangle\}$;
  $Type(w) := type$, $SType(w) := sType$, $Status(w) :=$ unexpanded;
  $Label(w) := label$, $RFmls(w) := rFmls$;
  if $type =$ state **then** $ILConstraints(w) := \emptyset$;
  if $v \neq$ null *and* $Type(v) =$ state **then** $ELabels(v, w) := \{eLabel\}$;
  if $indRepl \neq$ null **then** $IndRepl(w) := indRepl$
  else if $sType =$ complex **then**
    foreach *individual* a *occurring in* $Label(w)$ **do** $IndRepl(w)(a) := a$;
return w;

The graph G consists of two layers: the layer of complex nodes and the layer of simple nodes. There are no edges from simple nodes to complex nodes. The edges from complex nodes to simple nodes are exactly the edges outgoing from complex states. That is, if $\langle v, w \rangle$ is an edge from a complex node v to a simple node w then $Type(v) =$ state, if $Type(v) =$ state and $\langle v, w \rangle \in E$ then $SType(w) =$ simple. Each complex node of G is like an eABox (more formally, its label is an eABox), which can be treated as a graph whose vertices are named individuals. On the other hand, a simple node of G stands for an unnamed individual. If e is a label of an edge from a complex state v to a simple node w then the triple $\langle v, e, w \rangle$ can be treated as an edge from the named individual $\pi_I(e)$ (an inner node in the graph representing v) to the unnamed individual corresponding to w, and that edge is via the roles from $\pi_R(e)$.

We will use also assertions of the form $a : (\preceq n\, s.C)$ and $a : (\succeq n\, s.C)$, where s is a numeric role. The difference between $a : (\preceq n\, s.C)$ and $a : (\leq n\, s.C)$ is that, for checking $a : (\preceq n\, s.C)$, we do not have to pay attention to assertions of the form $s(a, b)$ or $r(a, b)$ with r being a subrole of s. The aim for $a : (\succeq n\, s.C)$ is similar. We use $a : (\preceq n\, s.C)$ and $a : (\succeq n\, s.C)$ only as syntactic representations of some expressions, and do not provide semantics for them. We define

$$\texttt{FullLabel}(v) = Label(v) \cup RFmls(v) - \{\text{formulas of the form } a\colon (\preceq n\, s.C) \text{ or } a\colon (\succeq n\, s.C)\}.$$

We apply global caching: if $v_1, v_2 \in V$, $Label(v_1) = Label(v_2)$ and ($SType(v_1) = SType(v_2) =$ simple or ($SType(v_1) = SType(v_2) =$ complex and $Type(v_1) = Type(v_2)$)) then $v_1 = v_2$. Due to global caching, an edge outgoing from a state may have a number of labels as the result of merging edges. Creation of a new node or a new edge is done by Procedure `ConToSucc` (connect to a successor) given on page 10. This procedure creates a connection from a node v given as the first parameter to a node w with $Type(w)$, $SType(w)$, $Label(w)$, $RFmls(w)$, $IndRepl(w)$, $ELabels(v, w)$ specified by the remaining parameters.

We say that a node v may affect the status of the root ν if there exists a path consisting of nodes $v_0 = \nu, v_1, \ldots, v_{n-1}, v_n = v$ such that, for every $0 \leq i < n$, $Status(v_i)$ differs from open and closed, and if it is closed-wrt(U) then U is disjoint from $\{v_0, \ldots, v_i\}$. In that case, if $u \in \{v_1, \ldots, v_n\}$ then we say that v may affect the status of the root ν via a path through u.

From now on, let $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ be a knowledge base in NNF of the logic $\mathcal{SHOQ}$, with $\mathcal{A} \neq \emptyset$.[11] In this section we present a tableau calculus $C_{\mathcal{SHOQ}}$ for checking satisfiability of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$. A $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ is a rooted graph $G = \langle V, E, \nu \rangle$ constructed as follows.

[11] If $\mathcal{A}$ is empty, we can add $a : \top$ to it, where a is a special individual.

Initialization: Set $V := \emptyset$ and $E := \emptyset$. Then, create the root node by executing $\nu :=$ $\texttt{ConToSucc}(\mathsf{null}, \mathsf{non\text{-}state}, \mathsf{complex}, \mathit{label}, \emptyset, \mathsf{null}, \mathsf{null})$, where $\mathit{label} = \mathcal{A} \cup \{(a\!:\!C) \mid C \in \mathcal{T}$ and a is an individual occurring in $\mathcal{A}$ or $\mathcal{T}\}$.

Rules' Priorities and Expansion Strategies: The graph is then expanded by the following rules, which will be specified shortly:

(UPS) rules for updating statuses of nodes,
(US) unary static expansion rules,
(DN) a rule for dealing with nominals,
(NUS) a non-unary static expansion rule,
(FS) the forming-state rule,
(TP) a transitional partial-expansion rule,
(TF) a transitional full-expansion rule.

Each of the rules is parametrized by a node v. We say that a rule is *applicable* to v if it can be applied to v to make changes to the graph. The rule (UPS) has a higher priority than (US), which has a higher priority than the remaining rules in the list. If neither (UPS) nor (US) is applicable to any node, then choose a node v with status $\mathsf{unexpanded}$ or $\mathsf{p\text{-}expanded}$, choose the first rule applicable to v among the rules in the last five items of the above list, and apply it to v. Any strategy can be used for choosing v, but it is worth to choose v for expansion only when v may affect the status of the root ν of the graph. Note that the priorities of the rules are specified by the order in the above list, but the rules (UPS) and (US) are checked globally (technically, they are triggered immediately when possible), while the remaining rules are checked for a chosen node.

The construction of the graph ends when the root ν receives the status closed or open or when no more changes that may affect the status of ν can be made.[12] Theorem 4.7 states that the knowledge base $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ is satisfiable iff $\mathit{Status}(\nu) \neq \mathsf{closed}$.

4.2 Techniques for Dealing with Nominals

As usual, to deal with assertions of the form $a : \{b\}$ (resp. $a : \neg\{b\}$) we use the predicate $\doteq$ (resp. $\not\doteq$) and the replacement technique. Recall also that we use statuses of the form $\mathsf{closed\text{-}wrt}(U)$ in order to be able to apply global caching in the presence of nominals. Updating statuses of nodes is defined appropriately. Our other techniques for dealing with nominals are described below.

Suppose v is a simple node with $\mathit{Status}(v) \notin \{\mathsf{closed}, \mathsf{open}\}$ and $\{a\} \in \mathit{Label}(v)$, a complex state u is an ancestor of v, and v may affect the status of the root ν via a path through u. Let u_0 be a predecessor of u. The node u_0 has only u as a successor and it was expanded by the forming-state rule. There are three cases:

- If, for every $C \in \mathit{Label}(v)$, the formula obtained from $a : C$ by replacing every individual b with $\mathit{IndRepl}(u)(b)$, when $\mathit{IndRepl}(u)(b)$ is defined, belongs to $\texttt{FullLabel}(u)$, then v is "consistent" with u.
- If there exists $C \in \mathit{Label}(v)$ such that the formula obtained from $a\!:\!\overline{C}$ (where $\overline{C}$ is the negation of C in NNF) by replacing every individual b with $\mathit{IndRepl}(u)(b)$, when $\mathit{IndRepl}(u)(b)$ is defined, belongs to $\texttt{FullLabel}(u)$, then v is "inconsistent" with u. In this case, if $\mathit{Status}(v)$ is of the form $\mathsf{closed\text{-}wrt}(U)$ then we update it to $\mathsf{closed\text{-}wrt}(U \cup \{u\})$, else we update it to $\mathsf{closed\text{-}wrt}(\{u\})$.

[12] That is, ignoring nodes that are unreachable from ν via a path without nodes with status closed or open, no more changes can be made to the graph.

– In the remaining case, the node u is "incomplete" w.r.t. v, which means that the expansion of u_0 was not appropriate. Thus, we delete the edge $\langle u_0, u\rangle$ and re-expand u_0 by an appropriate "or"-branching.

For dealing with interaction between number restrictions and nominals, to guarantee that every nominal represents a singleton set and a named individual cannot be cloned we use concepts of the form $\leq 1\, r.\{a\}$ and assertions of the form $a : \leq 1\, r.\{b\}$ that are *relevant* w.r.t. the TBox $\mathcal{T}$ and the label of the considered node. One can define this relation to be the full one (i.e., so that such formulas are always relevant). However, to increase efficiency we define this notion as follows.

Let X be a set of formulas. We say that a formula φ *occurs positively at the modal depth* 0 *in* X if there exist $\psi \in X$ and an occurrence of φ in ψ that is not in the scope of $\neg$, $\exists r$, $\forall r$, $\geq r\, n$, $\leq r\, n$ for any $r \in \mathbf{R}$. (Recall that formulas are in NNF.)

Definition 4.3. We say that a concept $\leq 1\, r.\{a\}$ is *relevant* w.r.t. a TBox $\mathcal{T}$ and a set X of concepts if the following conditions hold:

– either of the following conditions holds:
  • some concept $\geq m\, s.C$ with $m \geq 2$ and $s \sqsubseteq_{\mathcal{R}} r$ occurs positively at the modal depth 0 in X – in this case, let $s_1 = s_2 = s$ and $C_1 = C_2 = C$,
  • some different concepts C'_1 and C'_2 occur positively at the modal depth 0 in X and each C'_i is of the form $\exists s_i.C_i$ or $\geq 1\, s_i.C_i$ with $s_i \sqsubseteq_{\mathcal{R}} r$;
– one of the following conditions holds:
  • the nominal $\{a\}$ occurs positively at the modal depth 0 in $\mathcal{T}$ or $\{C_1\}$ or $\{C_2\}$,
  • some concept $\forall r'.D$ satisfying $s_1 \sqsubseteq r'$ and $s_2 \sqsubseteq r'$ occurs positively at the modal depth 0 in $\mathcal{T}$ and the nominal $\{a\}$ occurs positively at the modal depth 0 in $\{D\}$,
  • some concept $\leq n\, r'.D$ satisfying $s_1 \sqsubseteq r'$ and $s_2 \sqsubseteq r'$ occurs positively at the modal depth 0 in $\mathcal{T}$ and either $\{a\}$ or $\neg\{a\}$ occurs positively at the modal depth 0 in $\{D\}$.

A concept $a : \leq 1\, r.\{b\}$ is *relevant* w.r.t. a TBox $\mathcal{T}$ and a set X of eABox assertions if the concept $\leq 1\, r.\{b\}$ is relevant w.r.t. $\mathcal{T}$ and the set $\{C \mid a{:}C \in X\}$. □

Note that every interpretation $\mathcal{I}$ always validates $\leq 1\, r.\{a\}$ (i.e., $(\leq 1\, r.\{a\})^{\mathcal{I}} = \Delta^{\mathcal{I}}$) and satisfies $a : \leq 1\, r.\{b\}$ (i.e., $\mathcal{I} \models a : \leq 1\, r.\{b\}$).

4.3 Illustrative Examples

Before specifying the tableau rules in detail, we present simple examples to illustrate some ideas (but not all aspects) of our method. Despite that these examples refer to the tableau rules, we choose this place for presenting them because the examples are quite intuitive and the reader can catch the ideas of our method without knowing the detailed rules. He or she can always consult the rules in the next subsection.

Example 4.4. Let us construct a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$, where

$$\mathcal{A} = \{a{:}A,\ a{:}\exists r.\exists r.(A \sqcup \{a\}),\ a{:}\geq 3\, r.\forall r.\neg A,\ a{:}\forall r.B,\ a{:}\leq 3\, r.B,\\ r(a,b),\ b{:}\forall r.\neg A,\ b{:}(\forall r.(\neg A \sqcap \neg\{a\}) \sqcup \neg B)\},$$

$\mathcal{R} = \emptyset$ and $\mathcal{T} = \emptyset$. An illustration is presented in Figure 1.

At the beginning, the graph has only the root ν which is a complex non-state with $\mathit{Label}(\nu) = \mathcal{A}$. Since $\{a : \forall r.B, r(a,b)\} \subset \mathit{Label}(\nu)$, applying a unary static expansion rule to ν, we connect it to a new complex non-state v_1 with $\mathit{Label}(v_1) = \mathit{Label}(\nu) \cup \{b{:}B\}$.

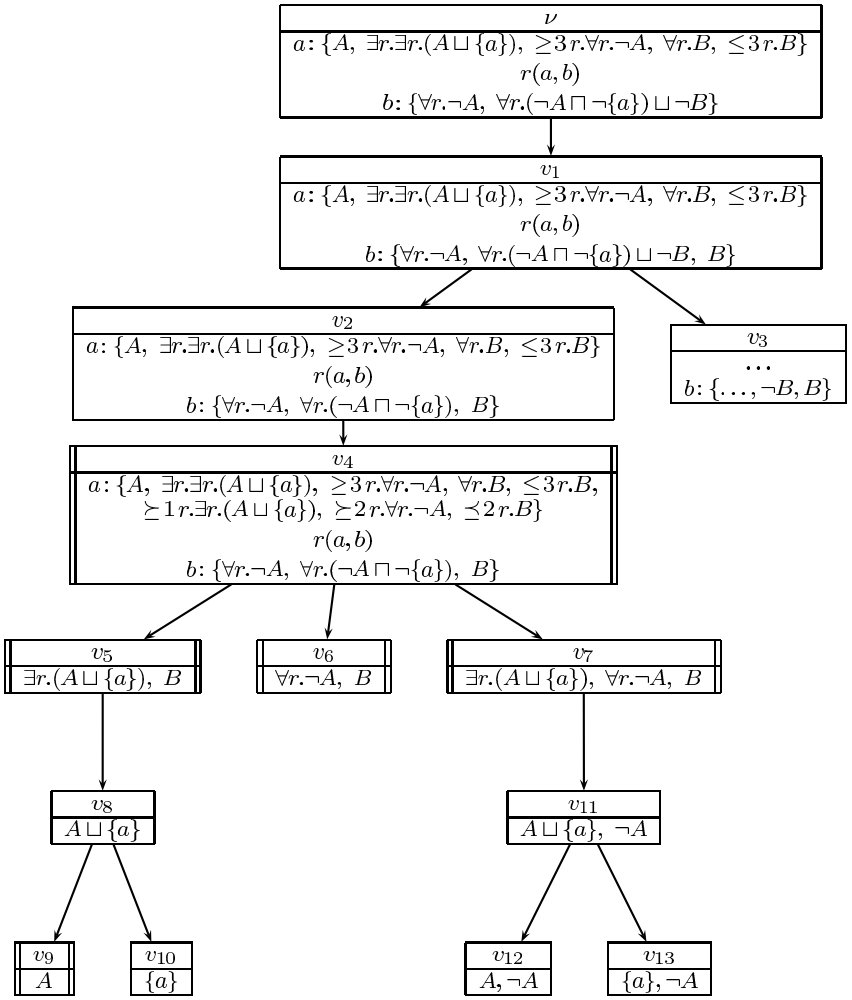

Fig. 1. An illustration of the tableau described in Example 4.4. The marked nodes v_4 – v_7 and v_9 are states. The nodes ν, v_1 – v_4 are complex nodes, the remaining are simple nodes. In each node, we display the formulas of its label.

Since $b\!:\!(\forall r.(\neg A \sqcap \neg\{a\}) \sqcup \neg B) \in \mathit{Label}(v_1)$, applying the non-unary static expansion rule to v_1, we connect it to new complex non-states v_2 and v_3 with

$$\mathit{Label}(v_2) = \mathit{Label}(v_1) - \{b\!:\!(\forall r.(\neg A \sqcap \neg\{a\}) \sqcup \neg B)\} \cup \{b\!:\!\forall r.(\neg A \sqcap \neg\{a\})\}$$
$$\mathit{Label}(v_3) = \mathit{Label}(v_1) - \{b\!:\!(\forall r.(\neg A \sqcap \neg\{a\}) \sqcup \neg B)\} \cup \{b\!:\!\neg B\}.$$

Since both $b : B$ and $b : \neg B$ belong to $\mathit{Label}(v_3)$, the node v_3 receives the status closed. Applying the forming-state rule to v_2, we connect it to a new complex state v_4 with

$$\mathit{Label}(v_4) = \mathit{Label}(v_2) \cup \{a\!:\!\succeq 1\, r.\exists r.(A \sqcup \{a\}),\ a\!:\!\succeq 2\, r.\forall r.\neg A,\ a\!:\!\preceq 2\, r.B\}.$$

The assertion $a\!:\!\succeq 1\, r.\exists r.(A \sqcup \{a\}) \in \mathit{Label}(v_4)$ is due to $a\!:\!\exists r.\exists r.(A \sqcup \{a\}) \in \mathit{Label}(v_2)$ and the fact that the negation of $b\!:\!\exists r.(A \sqcup \{a\})$ in NNF belongs to $\mathit{Label}(v_2)$ (notice that $r(a,b) \in \mathit{Label}(v_2)$). The assertion $a : \succeq 2\, r.\forall r.\neg A \in \mathit{Label}(v_4)$ is due to $a : \geq 3\, r.\forall r.\neg A \in \mathit{Label}(v_2)$ and the fact that $\{r(a,b),\ b\!:\!\forall r.\neg A\} \subset \mathit{Label}(v_2)$. Similarly, the assertion $a : \preceq 2\, r.B \in \mathit{Label}(v_4)$ is due to $a\!:\!\leq 3\, r.B \in \mathit{Label}(v_2)$ and the fact $\{r(a,b), b\!:\!B\} \subset \mathit{Label}(v_2)$.

As r is a numeric role, applying the transitional partial-expansion rule[13] to v_4, we just change the status of v_4 to p-expanded. After that, applying the transitional full-expansion

[13] which is used for making transitions via non-numeric roles

rule to v_4, we connect it to new simple non-states v_5, v_6, v_7, using the edge label $e = \langle \mathsf{checkingFeasibility}, \{r\}, a\rangle$, such that $\mathit{Label}(v_5) = \{\exists r.(A \sqcup \{a\}),\ B\}$, $\mathit{Label}(v_6) = \{\forall r.\neg A,\ B\}$, $\mathit{Label}(v_7) = \{\exists r.(A \sqcup \{a\}),\ \forall r.\neg A,\ B\}$. The creation of v_5 is caused by $a : \geq 1\, r.\exists r.(A \sqcup \{a\}) \in \mathit{Label}(v_4)$, while the creation of v_6 is caused by $a : \geq 1\, r.\forall r.\neg A$. The node v_7 results from merging v_5 and v_6. Furthermore, $\mathit{ILConstraints}(v_4)$ consists of $x_{v_i,e} \geq 0$, for $5 \leq i \leq 7$, and

$$\begin{aligned} x_{v_5,e} + x_{v_7,e} &\geq 1 \\ x_{v_6,e} + x_{v_7,e} &\geq 2 \\ x_{v_5,e} + x_{v_6,e} + x_{v_7,e} &\leq 2. \end{aligned}$$

Applying the forming-state rule to v_5, the type of this node is changed from non-state to state. Next, applying the transitional partial-expansion rule to v_5, its status is changed to p-expanded. Then, applying the transitional full-expansion rule to v_5, we connect v_5 to a new simple non-state v_8 with $\mathit{Label}(v_8) = \{A \sqcup \{a\}\}$ using the edge label $e' = \langle \mathsf{checkingFeasibility}, \{r\}, \mathsf{null}\rangle$ and set $\mathit{ILConstraints}(v_5) = \{x_{v_8,e'} \geq 0, x_{v_8,e'} \geq 1\}$.

Applying the non-unary static expansion rule to v_8, we connect it to new simple non-states v_9 and v_{10} with $\mathit{Label}(v_9) = \{A\}$ and $\mathit{Label}(v_{10}) = \{\{a\}\}$. The status of v_9 is then changed to open, which causes the statuses of v_8 and v_5 to be updated to open. The node v_{10} is not expanded as it does not affect the status of the root node ν.

Applying the forming-state rule to v_6, the type of this node is changed from non-state to state. Next, applying the transitional partial-expansion rule and then the transitional full-expansion rule to v_6, its status is changed to f-expanded. The status of v_6 is then updated to open.

Applying the forming-state rule to v_7, the type of this node is changed from non-state to state. Next, applying the transitional partial-expansion rule to v_7, its status is changed to p-expanded. Then, applying the transitional full-expansion rule to v_7, we connect v_7 to a new simple non-state v_{11} with $\mathit{Label}(v_{11}) = \{A \sqcup \{a\},\ \neg A\}$ using the mentioned edge label e' and set $\mathit{ILConstraints}(v_7) = \{x_{v_{11},e'} \geq 0,\ x_{v_{11},e'} \geq 1\}$.

Applying the non-unary static expansion rule to v_{11}, we connect it to new simple non-states v_{12} and v_{13} with $\mathit{Label}(v_{12}) = \{A, \neg A\}$ and $\mathit{Label}(v_{13}) = \{\{a\}, \neg A\}$. The status of v_{12} is then changed to closed. Since $a : A \in \mathit{Label}(v_4)$, the status of v_{13} is updated to closed-wrt$(\{v_4\})$, which causes the status of v_{11} to be updated also to closed-wrt$(\{v_4\})$. As the set $\mathit{ILConstraints}(v_7) \cup \{x_{v_{11},e'} = 0\}$ is infeasible, the status of v_7 is updated to closed-wrt$(\{v_4\})$. Next, as the set $\mathit{ILConstraints}(v_4) \cup \{x_{v_7,e} = 0\}$ is infeasible, the status of v_4 is first updated to closed-wrt$(\{v_4\})$ and then to closed. After that, the statuses of v_2, v_1, ν are sequentially updated to closed. Thus, we conclude that the knowledge base $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ is unsatisfiable. □

Example 4.5. Let us modify Example 4.4 by deleting the assertion $a : A$ from the ABox. That is, we are now constructing a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$, where

$$\begin{aligned} \mathcal{A} = \{&a : \exists r.\exists r.(A \sqcup \{a\}),\ a : \geq 3\, r.\forall r.\neg A,\ a : \forall r.B,\ a : \leq 3\, r.B, \\ &r(a,b),\ b : \forall r.\neg A,\ b : (\forall r.(\neg A \sqcap \neg\{a\}) \sqcup \neg B)\}, \end{aligned}$$

$\mathcal{R} = \emptyset$ and $\mathcal{T} = \emptyset$. The first stage of the construction is similar to the one of Example 4.4, up to the step of updating the status of v_{12} to closed. This stage is illustrated in Figure 2, which is similar to Figure 1 except that the labels of the nodes ν and v_1 – v_4 do not contain $a : A$. The continuation is described below and illustrated by Figure 3.

Since $\mathit{Label}(v_{13}) = \{\{a\}, \neg A\}$, applying the rule for dealing with nominals to v_{13}, we delete the edge $\langle v_2, v_4\rangle$ (from E) and re-expand v_2 by connecting it to new complex non-states v_{14} and v_{15} with $\mathit{Label}(v_{14}) = \mathit{Label}(v_2) \cup \{a : \neg A\}$ and $\mathit{Label}(v_{15}) = \mathit{Label}(v_2) \cup \{a : A\}$ as shown in Figure 3. The status of v_{13} is updated to blocked. The node v_4 is not deleted, but we do not display it in Figure 3.

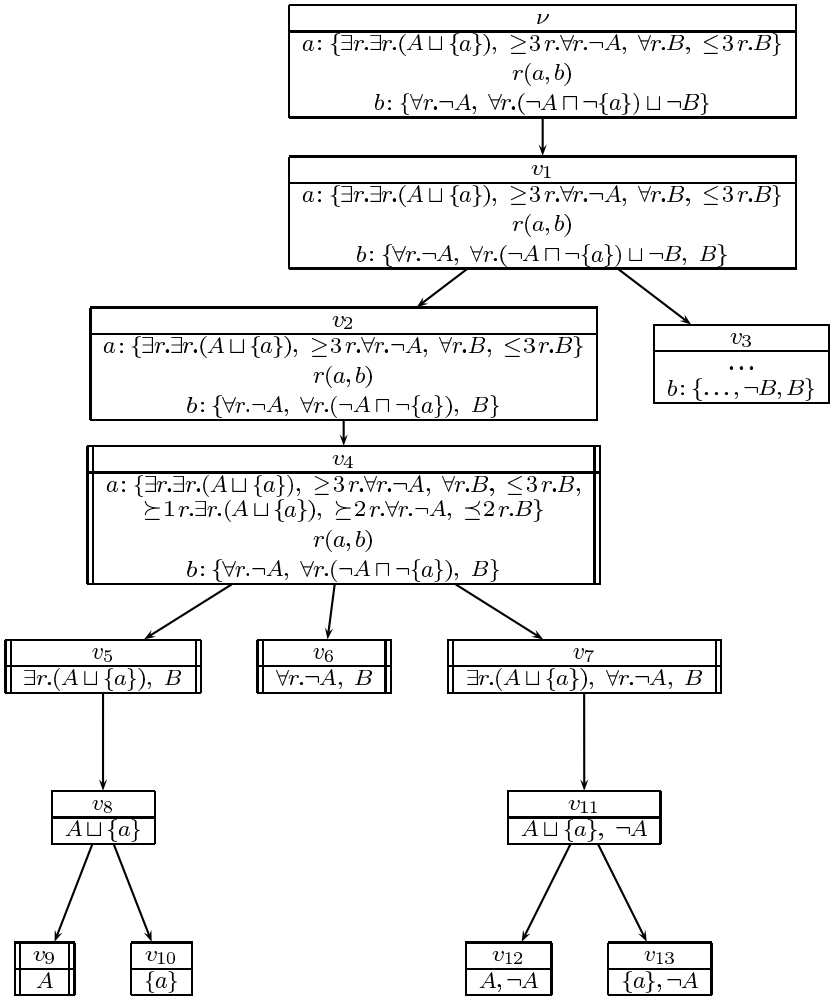

Fig. 2. An illustration for Example 4.5 – Part I.

Applying the forming-state rule to v_{14} we connect it to a new complex state v_{16}. The label of v_{16} is computed using $\mathit{Label}(v_{14})$ in a similar way as in Example 4.4 when computing $\mathit{Label}(v_4)$.

Applying the transitional partial-expansion rule to v_{16} we change its status to p-expanded. After that, applying the transitional full-expansion rule to v_{16} we connect it to the existing nodes v_5, v_6, v_7 using the edge label $e = \langle \mathsf{checkingFeasibility}, \{r\}, a\rangle$. The set $\mathit{ILConstraints}(v_{16})$ is the same as $\mathit{ILConstraints}(v_4)$.

Applying the forming-state rule to v_{15} we connect it to a new complex state v_{17}. The label of v_{17} is computed using $\mathit{Label}(v_{15})$ in a similar way as in Example 4.4 when computing $\mathit{Label}(v_4)$.

The expansion of v_{17} is similar to the expansion of v_{16}. The set $\mathit{ILConstraints}(v_{17})$ is the same as $\mathit{ILConstraints}(v_{16})$ and $\mathit{ILConstraints}(v_4)$. Analogously to updating the statuses of the nodes v_{13}, v_{11}, v_7 in Example 4.4 to closed-wrt($\{v_4\}$), the statuses of v_{13}, v_{11}, v_7 are updated to closed-wrt($\{v_{17}\}$). Next, as $\mathit{ILConstraints}(v_{17}) \cup \{x_{v_7,e} = 0\}$ is infeasible, the status of v_{17} is first updated to closed-wrt($\{v_{17}\}$) and then to closed. After that, the status of v_{15} is also updated to closed. As no more changes that may affect the status of ν can be made and $\mathit{Status}(\nu) \neq$ closed, we conclude that the knowledge base $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ is satisfiable. □

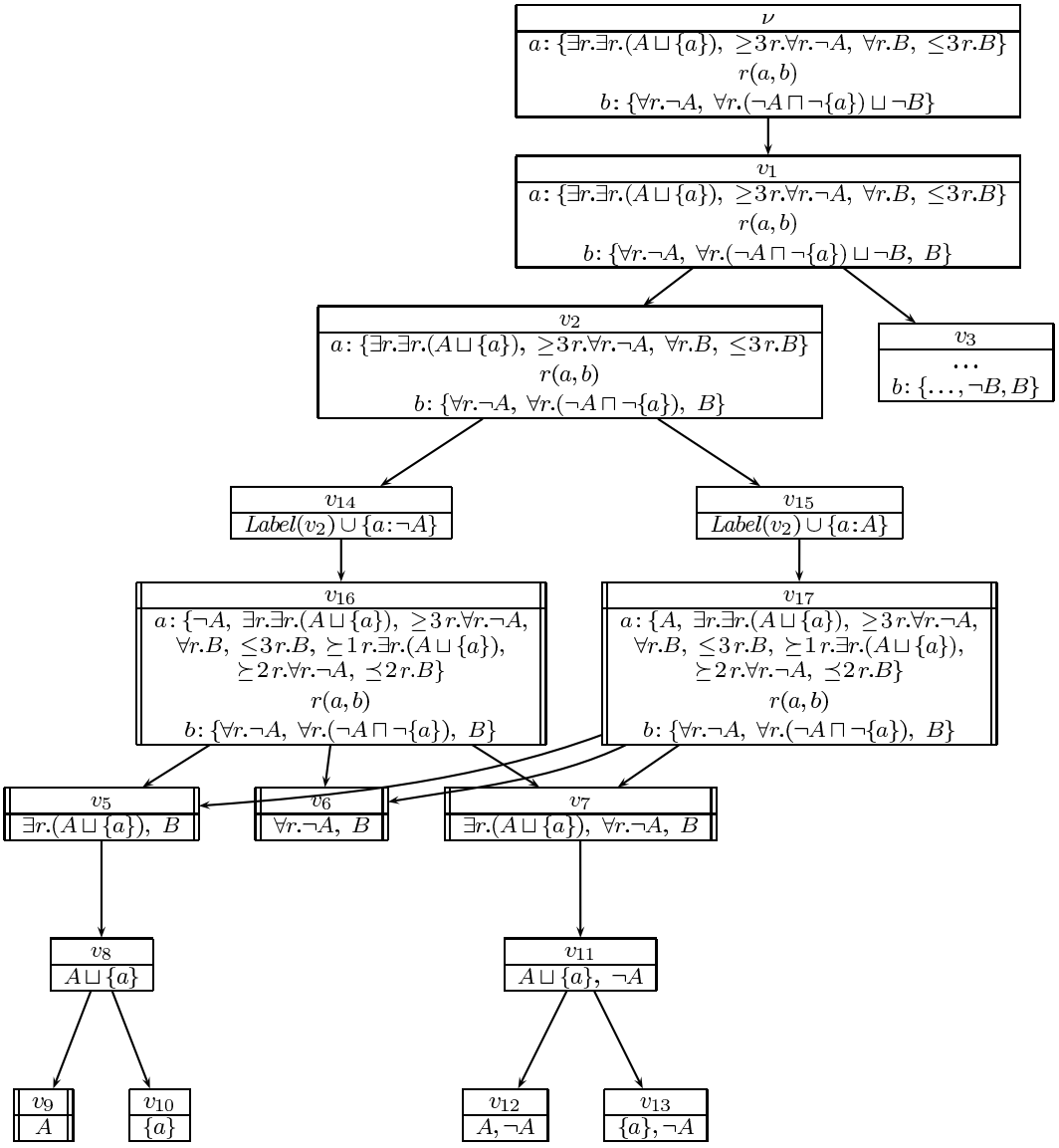

Fig. 3. An illustration for Example 4.5 – Part II.

4.4 Tableau Rules

In this subsection we formally specify the tableau rules of our calculus $C_{\mathcal{SHOQ}}$. We also give explanations for them. They are informal and should be understood in the context of the described rule.

We will use the auxiliary procedure $\texttt{SetClosedWrt}(v, u)$ defined as follows: "if $\mathit{Status}(v)$ is of the form $\mathsf{closed\text{-}wrt}(U)$ then $\mathit{Status}(v) := \mathsf{closed\text{-}wrt}(U \cup \{u\})$, else $\mathit{Status}(v) := \mathsf{closed\text{-}wrt}(\{u\})$". This procedure updates the status of v to reflect that v is closed w.r.t. u.

The Rules for Updating Statuses of Nodes:

($\mathbf{UPS}_1$) The first rule is as follows:

1. if one of the following conditions holds:
   (a) there exists $\alpha{:}\bot \in \mathit{Label}(v)$ or $\{\varphi, \overline{\varphi}\} \subseteq \texttt{FullLabel}(v)$,
   (b) there exists $a \neq a \in \mathit{Label}(v)$,

(c) $\mathit{Type}(v) = \mathsf{non\text{-}state}$, $a : (\leq n\, s.C) \in \mathit{Label}(v)$ and there are $b_0, \ldots, b_n \in \mathbf{I}$ such that, for all $0 \leq i, j \leq n$ with $i \neq j$, $\{s(a, b_i), b_i : C\} \subseteq \mathtt{FullLabel}(v)$ and $\{b_i \neq b_j, b_j \neq b_i\} \cap \mathit{Label}(v) \neq \emptyset$,
(d) $\mathit{Status}(v) = \mathsf{closed\text{-}wrt}(U)$ and $v \in U$,
then $\mathit{Status}(v) := \mathsf{closed}$
2. else if $\mathit{Type}(v) = \mathsf{state}$, $\mathit{Status}(v) = \mathsf{f\text{-}expanded}$ and v has no successors then $\mathit{Status}(v) := \mathsf{open}$.

Explanation 1 Informally, closed means "unsatisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$", open means "satisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$", and closed-wrt(U) means "unsatisfiable w.r.t. $\mathcal{R}$, $\mathcal{T}$ and any node from U". The above rule is thus intuitive. For a formal characterization of the statuses closed and closed-wrt(U), we refer the reader to Lemma A.4 (on page 27). □

(UPS$_2$) If $\mathit{SType}(v) = \mathsf{simple}$, $\mathit{Status}(v) \notin \{\mathsf{closed}, \mathsf{open}\}$ and $\mathit{Label}(v)$ contains a concept $\{a\}$ then
if there exist $u \in V$ and $C \in \mathit{Label}(v)$ such that
– $\mathit{Type}(u) = \mathsf{state} \wedge \mathit{SType}(u) = \mathsf{complex}$,
– v may affect the status of the root ν via a path through u,
– the assertion obtained from $a : \overline{C}$ by replacing every individual b by $\mathit{IndRepl}(u)(b)$ when $\mathit{IndRepl}(u)(b)$ is defined belongs to $\mathtt{FullLabel}(u)$
then $\mathtt{SetClosedWrt}(v, u)$.

Explanation 2 This rule deals with nominals. If u is a complex state and v is a simple node such that $\mathit{Label}(v)$ contains a nominal $\{a\}$ and v may affect the status of the root ν via a path through u then, when considering u for constructing a model for the considered knowledge base, the simple node v should be merged with the named individual a in the complex node u. If such a merging causes inconsistency then v is closed w.r.t. u and we update $\mathit{Status}(v)$ accordingly. □

(UPS$_3$) This rule states that, if v is a predecessor of a node w then, whenever the status of w changes to closed, closed-wrt(...) or open, the status of v should be updated (as soon as possible by using a priority queue of tasks). The update is done by one of the following subrules:
1. If $\mathit{Type}(v) = \mathsf{non\text{-}state}$ and $\mathit{Status}(v) \notin \{\mathsf{unexpanded}, \mathsf{closed}, \mathsf{open}\}$ then
(a) if some successor of v received status open then $\mathit{Status}(v) := \mathsf{open}$
(b) else if all successors of v have status closed then $\mathit{Status}(v) := \mathsf{closed}$
(c) else if every successor of v has status closed or closed-wrt(...) then:
i. let $w_1, \ldots, w_k$ be all the successors of v such that, for $1 \leq i \leq k$, $\mathit{Status}(w_i)$ is of the form closed-wrt(U_i), and let $U = \bigcap_{1 \leq i \leq k} U_i$
ii. for each $u \in U$ do $\mathtt{SetClosedWrt}(v, u)$.
2. If $\mathit{Type}(v) = \mathsf{state}$, $\mathit{Status}(v) \notin \{\mathsf{unexpanded}, \mathsf{closed}, \mathsf{open}\}$ and a successor w of v received status closed then
(a) if there exists $e \in \mathit{ELabels}(v, w)$ such that $\pi_T(e) = \mathsf{testingClosedness}$ then $\mathit{Status}(v) := \mathsf{closed}$
(b) else
– for each $e \in \mathit{ELabels}(v, w)$ such that $\pi_T(e) = \mathsf{checkingFeasibility}$ do add the constraint $x_{w,e} = 0$ to $\mathit{ILConstraints}(v)$
– if $\mathit{ILConstraints}(v)$ is infeasible then $\mathit{Status}(v) := \mathsf{closed}$.
3. If $\mathit{Type}(v) = \mathsf{state}$, $\mathit{Status}(v) \notin \{\mathsf{unexpanded}, \mathsf{closed}, \mathsf{open}\}$, a successor w of v received status closed-wrt(U), and v may affect the status of the root ν via a path through $u \in U$ then
(a) if there exists $e \in \mathit{ELabels}(v, w)$ such that $\pi_T(e) = \mathsf{testingClosedness}$ then $\mathtt{SetClosedWrt}(v, u)$

(b) else
- let $\langle w_1, e_1\rangle, \ldots, \langle w_k, e_k\rangle$ be all the pairs such that, for $1 \leq i \leq k$, w_i is a successor of v, $Status(w_i)$ is of the form $\mathsf{closed\text{-}wrt}(U_i)$ with $u \in U_i$, $e_i \in ELabels(v, w_i)$, and $\pi_T(e_i) = \mathsf{checkingFeasibility}$
- if $ILConstraints(v) \cup \{x_{w_i,e_i} = 0 \mid 1 \leq i \leq k\}$ is infeasible then $\mathtt{SetClosedWrt}(v, u)$.

4. If
- $Type(v) = \mathsf{state} \wedge Status(v) = \mathsf{f\text{-}expanded}$,
- every successor w of v with some $e \in ELabels(v, w)$ having $\pi_T(e) = \mathsf{testingClosedness}$ has status open, and
- $ILConstraints(v) \cup \{x_{w,e} = 0 \mid \langle v, w\rangle \in E,\ Status(w) \neq \mathsf{open},\ e \in ELabels(v, w)$ and $\pi_T(e) = \mathsf{checkingFeasibility}\}$ is feasible

then $Status(v) := \mathsf{open}$.

Explanation 3 For simplicity of understanding, one can first consider the case without nominals and statuses $\mathsf{closed\text{-}wrt}(\ldots)$. A non-state is like an "or"-node, whose status is the disjunction of the statuses of its successors, treating open as true and closed as false. A state is more sophisticated than an "and"-node. The status of a state v is different from closed iff the following conditions hold:
- for all successors w of v, if there exists $e \in ELabels(v, w)$ with $\pi_T(e) = \mathsf{testingClosedness}$ then $Status(w) \neq \mathsf{closed}$,
- $ILConstraints(v) \cup \{x_{w,e} = 0 \mid \langle v, w\rangle \in E,\ Status(w) = \mathsf{closed},\ e \in ELabels(v, w)$ and $\pi_T(e) = \mathsf{checkingFeasibility}\}$ is feasible.

The subrule 2 updates the status of a state v according to the above observation. The status open is a special case of being different from closed, which can be detected earlier, and the subrule 4 is defined appropriately, reflecting that observation.

Recall that $\mathsf{closed\text{-}wrt}(U)$ means closed w.r.t. any node $u \in U$, and such statuses are used for dealing with nominals. In the case with nominals and statuses $\mathsf{closed\text{-}wrt}(\ldots)$, for simplicity of understanding, one can imagine the traditional approach that does not use global caching but uses backtracking for dealing with "or"-branchings. With that approach, each node has at most one ancestor node u that is a complex state, and a status $\mathsf{closed\text{-}wrt}(\ldots)$ behaves similarly to the status closed. Our approach uses global caching and deals with nominals, and we use statuses $\mathsf{closed\text{-}wrt}(\ldots)$ in appropriate way to simulate the status closed. □

The Unary Static Expansion Rules:

($\mathbf{US}_1$) If $Type(v) = \mathsf{non\text{-}state}$ and $Status(v) = \mathsf{unexpanded}$ then

1. let $X = RFmls(v) \cup \{(\alpha : C) \in Label(v) \mid C$ is of the form $D \sqcap D'$ or $\geq 0\, s.D$ or $\leq 0\, s.D\} \cup \{a : \neg\{b\} \in Label(v)\}$
2. let $label = Label(v) \cup \{(\alpha : D), (\alpha : D') \mid \alpha : (D \sqcap D') \in Label(v)\}$
$\cup\ \{\alpha : \forall s.\overline{D} \mid\ (\alpha : \leq 0\, s.D) \in Label(v)\}$
$\cup\ \{\alpha : \forall r.D \mid \alpha : \forall s.D \in Label(v)$ and $r \sqsubseteq_{\mathcal{R}} s\}$
$\cup\ \{s(a, b) \mid r(a, b) \in Label(v)$ and $r \sqsubseteq_{\mathcal{R}} s\}$
$\cup\ \{b : D \mid \{a : \forall r.D, r(a, b)\} \subseteq Label(v)\}$
$\cup\ \{b : \forall r.D \mid \{a : \forall r.D, r(a, b)\} \subseteq Label(v)$ and $trans_{\mathcal{R}}(r)\}$
$\cup\ \{a \neq b \mid a : \neg\{b\} \in Label(v)\}$
$-\ X$
3. if $label - Label(v) \neq \emptyset$ then
   (a) $\mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, SType(v), label, X, IndRepl(v), \mathsf{null})$
   (b) $Status(v) := \mathsf{f\text{-}expanded}$.

Explanation 4 This rule makes a necessary expansion for a non-state v by connecting it to only one successor w which is a copy of w with intuitive changes like:

– if $\alpha\!:\!(D \sqcap D') \in \mathit{Label}(v)$ then $\alpha\!:\!(D \sqcap D')$ in $\mathit{Label}(w)$ is replaced by $\alpha\!:\!D$ and $\alpha\!:\!D'$ and we remember this by adding it to $\mathit{RFmls}(w)$;
– if $\{a\!:\!\forall r.D, r(a,b)\} \subseteq \mathit{Label}(v)$ then we add $b\!:\!D$ to $\mathit{Label}(w)$; and so on.

Note that $\mathit{Label}(w) - (\mathit{Label}(v) \cup \mathit{RFmls}(v)) \neq \emptyset$. That is, w contains some "new" formulas. □

(US$_2$) If $\mathit{Status}(v) = \mathsf{unexpanded}$ and $\mathit{Label}(v)$ contains $a\!:\!\{b\}$ then
1. let X be the set obtained from $\mathit{Label}(v) - \{a\!:\!\{b\}\}$ by replacing every occurrence of b not in $\doteq$ expressions by a
2. let Y be the set obtained from $\mathit{RFmls}(v)$ by replacing every occurrence of b by a
3. $w := \mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X \cup \{a \doteq b, b \doteq a\}, Y \cup \{a\!:\!\{a\}\}, \mathit{IndRepl}(v), \mathsf{null})$
4. $\mathit{IndRepl}(w)(b) := a$
5. for each $c \in \mathbf{I}$, if $\mathit{IndRepl}(v)(c) = b$ then $\mathit{IndRepl}(w)(c) := a$.
6. $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$.

Explanation 5 If v is an unexpanded complex node with $\mathit{Label}(v)$ containing $a\!:\!\{b\}$ then a and b should denote the same individual and we expand v by connecting it to only one successor w which is a copy of v with b replaced by a in an appropriate way. □

(US$_3$) If $\mathit{Type}(v) = \mathsf{non\text{-}state}$ and $\mathit{Status}(v) = \mathsf{unexpanded}$ then
1. if $\mathit{SType}(v) = \mathsf{simple}$ then let X be the set of all concepts of the form $\leq 1\, r.\{a\}$ that are relevant w.r.t. $\mathcal{T}$ and $\mathit{Label}(v)$, else let X be the set of all formulas of the form $a\!:\!\leq 1\, r.\{b\}$ that are relevant w.r.t. $\mathcal{T}$ and $\mathit{Label}(v)$
2. if $X - \mathit{Label}(v) \neq \emptyset$ then
$\mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathit{SType}(v), \mathit{Label}(v) \cup X, \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$.

Explanation 6 This rule deals with interaction between number restrictions and nominals. We want to guarantee that every nominal represents a singleton set and a named individual cannot be cloned. □

The Rule for Dealing with Nominals:

(DN) If $\mathit{SType}(v) = \mathsf{simple}$, $\mathit{Status}(v) \notin \{\mathsf{closed}, \mathsf{open}\}$ and $\mathit{Label}(v)$ contains $\{a\}$ then
1. for each complex state u such that v is not closed w.r.t. u (i.e., $\mathit{Status}(v)$ is not of the form $\mathsf{closed\text{-}wrt}(U)$ with $u \in U$) and v may affect the status of the root ν via a path through u, do
   (a) let $X = \{\varphi_1, \ldots, \varphi_k\}$ be the set obtained from $\{a\!:\!C \mid C \in \mathit{Label}(v)\}$ by replacing every individual b by $\mathit{IndRepl}(u)(b)$ when $\mathit{IndRepl}(u)(b)$ is defined
   (b) if $X \not\subseteq \mathtt{FullLabel}(u)$ then:
   for each predecessor u_0 of u do
      i. delete the edge $\langle u_0, u\rangle$ from E and its labels from $\mathit{ELabels}$
      ii. $\mathtt{ConToSucc}(u_0, \mathsf{non\text{-}state}, \mathsf{complex}, \mathit{Label}(u_0) \cup X, \mathit{RFmls}(u_0), \mathit{IndRepl}(u_0), \mathsf{null})$
      iii. for each $1 \leq i \leq k$ such that $\varphi_i \notin \mathtt{FullLabel}(u)$ do:
      $\mathtt{ConToSucc}(u_0, \mathsf{non\text{-}state}, \mathsf{complex}, \mathit{Label}(u_0) \cup \{\overline{\varphi_i}\}, \mathit{RFmls}(u_0), \mathit{IndRepl}(u_0), \mathsf{null})$
2. $\mathit{Status}(v) := \mathsf{blocked}$.

Explanation 7 If u is a complex state and v is a simple node such that $\mathit{Label}(v)$ contains a nominal $\{a\}$ and v may affect the status of the root ν via a path through u then, when considering u for constructing a model for the considered knowledge base, the simple node v should be merged with the named individual a in the complex node u. The case when such a merging causes inconsistency is dealt with by the rule (UPS$_2$) (with a higher priority). Consider the other case. The set $X = \{\varphi_1, \ldots, \varphi_k\}$ of assertions computed at the step 1a of the rule would be added to $\mathit{Label}(u)$. However, we do not want to modify labels of nodes.

In the case when $X \not\subseteq \texttt{FullLabel}(u)$, the label of u is "incomplete" and we re-expand every predecessor u_0 of u by deleting the edge $\langle u_0, u\rangle$ and connecting u_0 to $k+1$ successors, where the label of the successor number 0 extends $\mathit{Label}(u_0)$ with $X = \{\varphi_1, \ldots, \varphi_k\}$ and the label of the successor number i ($1 \leq i \leq k$) extends $\mathit{Label}(u_0)$ with $\overline{\varphi_i}$ (the negation of φ_i in NNF). This is like an on-demand cut. □

The Non-unary Static Expansion Rule:

(NUS) If $\mathit{Type}(v) = \mathsf{non\text{-}state}$ and $\mathit{Status}(v) = \mathsf{unexpanded}$ then

1. if $\alpha{:}(C \sqcup D) \in \mathit{Label}(v)$ and $\{\alpha{:}C, \alpha{:}D\} \cap \texttt{FullLabel}(v) = \emptyset$ then
   (a) let $X = \mathit{Label}(v) - \{\alpha{:}(C \sqcup D)\}$
   (b) let $Y = \mathit{RFmls}(v) \cup \{\alpha{:}(C \sqcup D)\}$
   (c) $\texttt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathit{SType}(v), X \cup \{\alpha{:}C\}, Y, \mathit{IndRepl}(v), \mathsf{null})$
   (d) $\texttt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathit{SType}(v), X \cup \{\alpha{:}D\}, Y, \mathit{IndRepl}(v), \mathsf{null})$
   (e) $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$

   Explanation 8 This subrule deals with syntactic branching on $\alpha{:}(C \sqcup D) \in \mathit{Label}(v)$. We expand v by connecting it to two successors w_1 and w_2, whose labels are the label of v with $\alpha{:}(C \sqcup D)$ replaced by $\alpha{:}C$ or $\alpha{:}D$, respectively. The formula $\alpha{:}(C \sqcup D)$ is put into both $\mathit{RFmls}(w_1)$ and $\mathit{RFmls}(w_2)$. The expansion is done only when both w_1 and w_2 have a larger $\texttt{FullLabel}$ than v. □

2. else if $\mathit{SType}(v) = \mathsf{complex}$, $s(a,b) \in \mathit{Label}(v)$ and
   – $\mathit{Label}(v)$ contains $a{:}(\leq n\, s.C)$, or
   – $\mathit{Label}(v)$ contains $a{:}(\geq n\, s.C)$ or $a{:}(\exists s.C)$, where s is a numeric role,
   and $\{b{:}C, b{:}\overline{C}\} \cap \texttt{FullLabel}(v) = \emptyset$ then
   (a) let $X = \mathit{Label}(v) \cup \{b{:}C\}$ and $X' = \mathit{Label}(v) \cup \{b{:}\overline{C}\}$
   (b) $\texttt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X, \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$
   (c) $\texttt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X', \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$
   (d) $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$

   Explanation 9 This subrule deals with the case when there is a lack of information about b for deciding how to satisfy the number restrictions about a. We want to have either $b : C$ or $b : \overline{C}$ in $\texttt{FullLabel}(v)$. So, we expand v by semantic branching: we connect it to two successors, one with label $\mathit{Label}(v) \cup \{b : C\}$ and the other with label $\mathit{Label}(v) \cup \{b{:}\overline{C}\}$. The expansion is done only when both the successors have a larger $\texttt{FullLabel}$ than v. □

3. else if $\mathit{SType}(v) = \mathsf{complex}$, $\{a{:}(\leq n\, s.C),\ s(a,b),\ s(a,b'),\ b{:}C,\ b'{:}C\} \subseteq \texttt{FullLabel}(v)$, $b \neq b'$ and $\{b \not\doteq b', b' \not\doteq b\} \cap \mathit{Label}(v) = \emptyset$ then[14]
   (a) let $X_1 = \mathit{Label}(v) \cup \{b \not\doteq b', b' \not\doteq b\}$ and let X_2 be the set obtained from $\mathit{Label}(v)$ by replacing every occurrence of b' not in $\doteq$ expressions by b
   (b) let Y be the set obtained from $\mathit{RFmls}(v)$ by replacing every occurrence of b' by b
   (c) $\texttt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X_1, \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$
   (d) $w := \texttt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X_2 \cup \{b \doteq b', b' \doteq b\}, Y, \mathit{IndRepl}(v), \mathsf{null})$
   (e) $\mathit{IndRepl}(w)(b') := b$
   (f) for each $c \in \mathbf{I}$, if $\mathit{IndRepl}(v)(c) = b'$ then $\mathit{IndRepl}(w)(c) := b$
   (g) $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$

   Explanation 10 This subrule deals with the case when there is a lack of information about whether b and b' denote the same individual for deciding how to satisfy the number restrictions about a. We expand v by semantic branching: either b and b' denote the same individual or they do not. Technically, we connect v to two successors with appropriate contents. □

[14] Fix a linear order between individual names. Then we can also assume that b is less than b' in that order.

4. else if $\mathit{SType}(v) = \mathsf{complex}$, $\{a : (\leq m\, r.C),\ r(a,b)\} \subseteq \mathit{Label}(v)$, $\mathit{Label}(v)$ contains $a\!:\!(\geq n\, s.D)$ or $a\!:\!\exists s.D$ with $s \sqsubseteq_{\mathcal{R}} r$, and $\{s(a,b), \neg s(a,b)\} \cap \mathit{Label}(v) = \emptyset$ then
   (a) let $X_1 = \mathit{Label}(v) \cup \{s(a,b)\}$ and $X_2 = \mathit{Label}(v) \cup \{\neg s(a,b)\}$
   (b) $\mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X_1, \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$
   (c) $\mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{complex}, X_2, \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$
   (d) $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$.

Explanation 11 This subrule deals with the case when there is a lack of information for deciding how to satisfy the number restrictions about a. We want to decide whether b is an s-successor of a or not. So, we expand v by semantic branching: we connect it to two successors, one with label containing $s(a,b)$ and the other with label containing $\neg s(a,b)$. The expansion is done only when both the successors have a larger label than v. □

The Forming-State Rule:

(FS) If $\mathit{Type}(v) = \mathsf{non\text{-}state}$ and $\mathit{Status}(v) = \mathsf{unexpanded}$ then
1. if $\mathit{SType}(v) = \mathsf{simple}$ then $\mathit{Type}(v) := \mathsf{state}$ and $\mathit{ILConstraints}(v) := \emptyset$
2. else
   (a) set $X := \mathit{Label}(v)$
   (b) for each $a\!:\!(\leq n\, s.D) \in \mathit{Label}(v)$ do
      i. let $m = \sharp\{b \mid \{s(a,b), b\!:\!D\} \subseteq \mathtt{FullLabel}(v)\}$
      ii. add $a\!:\!(\preceq (n-m)\, s.D)$ to X
   (c) for each $(a\!:\!C) \in \mathit{Label}(v)$, where C is $\geq n\, s.D$ or $\exists s.D$ and s is a numeric role, do
      i. if $C = \exists s.D$ then let $n = 1$
      ii. let $m = \sharp\{b \mid \{s(a,b), b\!:\!D\} \subseteq \mathtt{FullLabel}(v)\}$
      iii. if $n > m$ then add $a\!:\!(\succeq (n-m)\, s.D)$ to X
   (d) $\mathtt{ConToSucc}(v, \mathsf{state}, \mathsf{complex}, X, \mathit{RFmls}(v), \mathit{IndRepl}(v), \mathsf{null})$
   (e) $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$.

Explanation 12 When the rules (UPS), (US), (DN) and (NUS) are not applicable to the non-state v, we apply this forming-state rule to v. If v is a complex node then we connect it to a complex state w. When computing contents for w we put into $\mathit{Label}(w)$ the requirements from $\mathit{Label}(v)$ after an appropriate modification that takes into account the assertions in $\mathit{Label}(v)$ that represent the relationship between named individuals. For example, if $a\!:\!(\leq n\, s.D) \in \mathit{Label}(v)$ and there are m pairwise different individuals $b_1, \ldots, b_m$ such that $\{s(a,b_i), b_i\!:\!D \mid 1 \leq i \leq m\} \subseteq \mathtt{FullLabel}(v)$ then we add to $\mathit{Label}(w)$ the requirements $a\!:\!(\preceq (n-m)\, s.D)$. Notice the use of $\preceq$ instead of $\leq$. Note that, since the rule (NUS) is not applicable to v, we must have that $(b_i \neq b_j) \in \mathit{Label}(v)$ for any pair $i \neq j$, and for any individual b such that $s(a,b) \in \mathit{Label}(v)$, either $b\!:\!D$ or $b\!:\!\overline{D}$ must belong to $\mathtt{FullLabel}(v)$. When expanding w we will not have to pay attention to the relationship between the individuals occurring in $\mathit{Label}(w)$.

If v is a simple node then we just change $\mathit{Type}(v)$ to state and initialize $\mathit{ILConstraints}(v)$ to $\emptyset$. Number restrictions about v are dealt with later by the transitional full-expansion rule.

The way of forming a state for a complex node v is more sophisticated (than for a simple node) because we may need to re-expand v later due to nominals (as done by the rule (DN) for u_0). □

The Transitional Partial-Expansion Rule:

(TP) If $\mathit{Type}(v) = \mathsf{state}$ and $\mathit{Status}(v) = \mathsf{unexpanded}$ then
1. for each $(\alpha\!:\!\exists r.D) \in \mathit{Label}(v)$, where r is a non-numeric role, do

(a) $X := \{D\} \cup \{D' \mid \alpha{:}\forall r.D' \in \mathit{Label}(v)\} \cup$
$\{\forall s.D' \mid \alpha{:}\forall s.D' \in \mathit{Label}(v), r \sqsubseteq_{\mathcal{R}} s \text{ and } \mathit{trans}_{\mathcal{R}}(s)\} \cup \mathcal{T}$
(b) $e\mathit{Label} := \langle \mathsf{testingCloseness}, \{s \mid r \sqsubseteq_{\mathcal{R}} s\}, \alpha\rangle$
(c) $\mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{simple}, X, \emptyset, \mathsf{null}, e\mathit{Label})$
2. $\mathit{Status}(v) := \mathsf{p\text{-}expanded}$.

Explanation 13 To realize a requirement $\alpha{:}\exists r.D$ at a state v, where r is a non-numeric role, we connect v to a new simple non-state w with appropriate contents as shown in the rule. □

The Transitional Full-Expansion Rule:

(TF) If $\mathit{Type}(v) = \mathsf{state}$ and $\mathit{Status}(v) = \mathsf{p\text{-}expanded}$ then
1. if $\mathit{SType}(v) = \mathsf{complex}$
then let $\Gamma = \mathit{Label}(v)$
else let $\Gamma = \mathit{Label}(v) \cup \{\preceq n\, r.D \mid \leq n\, r.D \in \mathit{Label}(v)\} \cup$
$\{\succeq n\, r.D \mid \geq n\, r.D \in \mathit{Label}(v)\} \cup$
$\{\succeq 1\, r.D \mid \exists r.D \in \mathit{Label}(v) \text{ and } r \text{ is a numeric role}\}$
2. $\mathcal{E} := \emptyset$, $\mathcal{E}' := \emptyset$
3. for each $(\alpha{:}\succeq n\, r.D) \in \Gamma$ do
(a) $X := \{s \mid r \sqsubseteq_{\mathcal{R}} s\}$
(b) $Y := \{D\} \cup \{D' \mid \alpha{:}\forall r.D' \in \Gamma\} \cup$
$\{\forall s.D' \mid \alpha{:}\forall s.D' \in \Gamma, r \sqsubseteq_{\mathcal{R}} s \text{ and } \mathit{trans}_{\mathcal{R}}(s)\} \cup \mathcal{T}$
(c) $\mathcal{E} := \mathcal{E} \cup \{\langle X, Y, \alpha\rangle\}$
4. for each $\alpha{:}(\preceq n\, r.C) \in \Gamma$ do
(a) for each $\langle X, Y, \alpha\rangle \in \mathcal{E}$ do
i. if $r \in X$ and $\{C, \overline{C}\} \cap Y = \emptyset$ then
$\mathcal{E}' := \mathcal{E}' \cup \{\langle X, Y \cup \{C\}, \alpha\rangle, \langle X, Y \cup \{\overline{C}\}, \alpha\rangle\}$
(i.e., $\langle X, Y, \alpha\rangle$ is replaced by $\langle X, Y \cup \{C\}, \alpha\rangle$ and $\langle X, Y \cup \{\overline{C}\}, \alpha\rangle$)
ii. else $\mathcal{E}' := \mathcal{E}' \cup \{\langle X, Y, \alpha\rangle\}$
(b) $\mathcal{E} := \mathcal{E}'$, $\mathcal{E}' := \emptyset$
5. repeat
for each $\alpha{:}(\preceq n\, r.C) \in \Gamma$, $\langle X, Y, \alpha\rangle \in \mathcal{E}$ and $\langle X', Y', \alpha\rangle \in \mathcal{E}$ such that $r \in X$, $C \in Y$, $r \in X'$, $C \in Y'$, $\langle X \cup X', Y \cup Y', \alpha\rangle \notin \mathcal{E}$ and $Y \cup Y'$ does not contain any pair of the form $\varphi, \overline{\varphi}$ do add $\langle X \cup X', Y \cup Y', \alpha\rangle$ to $\mathcal{E}$ (i.e., the merger of $\langle X, Y, \alpha\rangle$ and $\langle X', Y', \alpha\rangle$ is added to $\mathcal{E}$)
until no tuples were added to $\mathcal{E}$ during the last iteration
6. for each $\langle X, Y, \alpha\rangle \in \mathcal{E}$ do
$\mathtt{ConToSucc}(v, \mathsf{non\text{-}state}, \mathsf{simple}, Y, \emptyset, \mathsf{null}, \langle \mathsf{checkingFeasibility}, X, \alpha\rangle)$
7. $\mathit{ILConstraints}(v) := \{x_{w,e} \geq 0 \mid \langle v, w\rangle \in E,\ e \in \mathit{ELabels}(v, w) \text{ and } \pi_T(e) = \mathsf{checkingFeasibility}\}$
8. for each $(\alpha{:}C) \in \Gamma$ do
(a) if C is of the form $\succeq n\, r.D$ then add to $\mathit{ILConstraints}(v)$ the constraint $\sum\{x_{w,e} \mid \langle v, w\rangle \in E,\ e \in \mathit{ELabels}(v, w),\ \pi_T(e) = \mathsf{checkingFeasibility},\ r \in \pi_R(e),\ \pi_I(e) = \alpha,\ D \in \mathit{Label}(w)\} \geq n$
(b) if C is of the form $\preceq n\, r.D$ then add to $\mathit{ILConstraints}(v)$ the constraint $\sum\{x_{w,e} \mid \langle v, w\rangle \in E,\ e \in \mathit{ELabels}(v, w),\ \pi_T(e) = \mathsf{checkingFeasibility},\ r \in \pi_R(e),\ \pi_I(e) = \alpha,\ D \in \mathit{Label}(w)\} \leq n$
9. $\mathit{Status}(v) := \mathsf{f\text{-}expanded}$.

Explanation 14 Let Γ be the set computed at the step 1. It consists of the requirements to be realized for v. To satisfy a requirement $\varphi = (\alpha : \geq n\, r.C) \in \Gamma$ for v, one can first connect v to a successor w_φ using an edge label e specified by the tuple $\langle X, Y, \alpha\rangle$ computed at the step 3, where $\pi_T(e) = \mathsf{checkingFeasibility}$, $\pi_R(e) = X$, $\pi_I(e) = \alpha$ and Y represents $Label(w_\varphi)$, and then clone w_φ to create n successors for v (or only record the intention somehow). The label of w_φ contains only formulas necessary for realizing the requirement $\alpha : \exists r.C$ and related ones of the form $\alpha : \forall r'.C'$ in Γ. To satisfy requirements of the form $\alpha : \preceq n'\, r'.C'$ for v, where $r \sqsubseteq_{\mathcal{R}} r'$, we tend to use only copies of w_φ extended with either C' or $\overline{C'}$ (for easy counting) as well as the mergers of such extended nodes. So, we first start with the set $\mathcal{E}$ constructed at the step 3, which consists of tuples with information about successors to be created for v. We then modify $\mathcal{E}$ by taking necessary extensions of the nodes (see the step 4). After that, we continue modifying $\mathcal{E}$ by adding to it also appropriate mergers of nodes and edges (see the step 5). Successors for v are created at the step 6. The number of copies of a node w that are intended to be used as successors of v using an edge label e is represented by a variable $x_{w,e}$ (we will not actually create such copies). The case when w would be a named individual and cannot be cloned is dealt with by the rule (US_3) (see Explanation 6). The set $ILConstraints(v)$ consisting of appropriate constraints about such variables are set at the steps 7-8. □

4.5 Properties of $C_{\mathcal{SHOQ}}$-Tableaux

Define the size of a knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ to be the number of bits used for the usual sequential representation of KB. It is greater than the number of symbols occurring in KB. If N is the size of KB and $\leq n\, r.C$ or $\geq n\, r.C$ is a number restriction occurring in KB then:

- when numbers are coded in unary we have that $n \leq \mathsf{N}$,
- when numbers are coded in binary we have that $n \leq 2^{\mathsf{N}}$.

Lemma 4.6 (Complexity). *Let $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ be a knowledge base in NNF of the logic $\mathcal{SHOQ}$ and let N be the size of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$. Then a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ can be constructed in (at most) exponential time in N in the following cases:*

1. *numbers are coded in unary,*
2. *numbers are coded in binary and $n \leq \mathsf{N}$ for every concept $\leq n\, r.C$ occurring in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$,*
3. *numbers are coded in binary and $n \leq \mathsf{N}$ for every concept $\geq n\, r.C$ occurring in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$.* □

Theorem 4.7 (Soundness and Completeness). *Let $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ be a knowledge base in NNF of the logic $\mathcal{SHOQ}$ and $G = \langle V, E, \nu\rangle$ be an arbitrary $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$. Then $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ is satisfiable iff $Status(\nu) \neq \mathsf{closed}$.* □

See the Appendix for the proofs of the above lemma and theorem.

To check satisfiability of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ one can construct a $C_{\mathcal{SHOQ}}$-tableau for it, then return "no" when the root of the tableau has status closed, or "yes" in the other cases. We call this the *$C_{\mathcal{SHOQ}}$-tableau decision procedure*. The corollary given below immediately follows from Theorem 4.7 and Lemma 4.6.

Corollary 4.8. *The $C_{\mathcal{SHOQ}}$-tableau decision procedure has* ExpTime *complexity when numbers are coded in unary.* □

5 Conclusions

Recall that $\mathcal{SHIQ}$, $\mathcal{SHOQ}$, $\mathcal{SHIO}$ are the three most well-known expressive DLs with ExpTime complexity. (Due to the interaction between $\mathcal{I}$, $\mathcal{Q}$ and $\mathcal{O}$, the complexity of the DL $\mathcal{SHOIQ}$ is NExpTime-complete). In this paper, we have presented the first ExpTime tableau decision

procedure for checking satisfiability of a knowledge base in the DL $\mathcal{SHOQ}$ when numbers are coded in unary.

We applied Nguyen's method [18] of integer linear feasibility checking for dealing with number restrictions. This work differs from the work [18] in that nominals are allowed instead of inverse roles. Without inverse roles, global caching is used instead of global state caching to allow more cache hits. We used special techniques for dealing with nominals and their interaction with number restrictions.

Acknowledgments. This work was supported by Polish National Science Centre (NCN) under Grants No. 2011/01/B/ST6/02759 (for the first author) and 2011/02/A/HS1/00395 (for the second author).

References

1. B. Dunin-Kęplicz, L.A. Nguyen, and A. Szałas. Converse-PDL with regular inclusion axioms: A framework for MAS logics. *J. Applied Non-Classical Logics*, 21(1):61–91, 2011.
2. J. Faddoul and V. Haarslev. Algebraic tableau reasoning for the description logic SHOQ. *J. Applied Logic*, 8(4):334–355, 2010.
3. N. Farsiniamarj. Combining integer programming and tableau-based reasoning: a hybrid calculus for the description logic SHQ. Master's thesis, Concordia University, 2008.
4. R. Goré. Tableau methods for modal and temporal logics. In D'Agostino et al, editor, *Handbook of Tableau Methods*, pages 297–396. Kluwer, 1999.
5. R. Goré and L.A. Nguyen. A tableau system with automaton-labelled formulae for regular grammar logics. In B. Beckert, editor, *Proceedings of TABLEAUX 2005*, volume 3702 of *LNAI*, pages 138–152. Springer, 2005.
6. R. Goré and L.A. Nguyen. ExpTime tableaux with global caching for description logics with transitive roles, inverse roles and role hierarchies. In N. Olivetti, editor, *Proceedings of TABLEAUX 2007*, volume 4548 of *LNAI*, pages 133–148. Springer, 2007.
7. R. Goré and L.A. Nguyen. Analytic cut-free tableaux for regular modal logics of agent beliefs. In F. Sadri and K. Satoh, editors, *Proceedings of CLIMA VIII*, volume 5056 of *LNAI*, pages 268–287. Springer, 2008.
8. R. Goré and L.A. Nguyen. Exptime tableaux for ALC using sound global caching. *J. Autom. Reasoning*, 50(4):355–381, 2013.
9. R. Goré and F. Widmann. Sound global state caching for $\mathcal{ALC}$ with inverse roles. In M. Giese and A. Waaler, editors, *Proceedings of TABLEAUX 2009*, volume 5607 of *LNCS*, pages 205–219. Springer, 2009.
10. R. Goré and F. Widmann. Optimal and cut-free tableaux for propositional dynamic logic with converse. In J. Giesl and R. Hähnle, editors, *Proceedings of IJCAR 2010*, volume 6173 of *LNCS*, pages 225–239. Springer, 2010.
11. J. Hladik and J. Model. Tableau systems for SHIO and SHIQ. In *Proceedings of Description Logics'2004*, volume 104 of *CEUR Workshop Proceedings*, pages 168–177, 2004.
12. I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In P. Doherty, J. Mylopoulos, and C.A. Welty, editors, *Proc. of KR'2006*, pages 57–67. AAAI Press, 2006.
13. I. Horrocks and U. Sattler. Ontology reasoning in the SHOQ(D) description logic. In B. Nebel, editor, *Proceedings of IJCAI'2001*, pages 199–204. Morgan Kaufmann, 2001.
14. A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.
15. L.A. Nguyen. Analytic tableau systems and interpolation for the modal logics KB, KDB, K5, KD5. *Studia Logica*, 69(1):41–57, 2001.
16. L.A. Nguyen. Cut-free ExpTime tableaux for checking satisfiability of a knowledge base in the description logic $\mathcal{ALCI}$. In *Proceedings of ISMIS'2011*, volume 6804 of *LNAI*, pages 465–475. Springer, 2011.
17. L.A. Nguyen. A cut-free ExpTime tableau decision procedure for the description logic SHI. In *Proceedings of ICCCI'2011 (1)*, volume 6922 of *LNCS*, pages 572–581. Springer, 2011 (see also the long version `http://arxiv.org/abs/1106.2305`).
18. L.A. Nguyen. ExpTime tableaux for the description logic SHIQ based on global state caching and integer linear feasibility checking. arXiv:1205.5838, 2012.
19. L.A. Nguyen. A tableau method with optimal complexity for deciding the description logic SHIQ. In *Proceedings of ICCSAMA'2013*, volume 479 of *Studies in Computational Intelligence*, pages 331–342. Springer, 2013.
20. L.A. Nguyen and J. Golińska-Pilarek. An exptime tableau method for dealing with nominals and quantified number restrictions in deciding the description logic shoq. In *Proceedings of CS&P'2013*.
21. L.A. Nguyen and A. Szałas. ExpTime tableaux for checking satisfiability of a knowledge base in the description logic ALC. In N.T. Nguyen, R. Kowalczyk, and S.-M. Chen, editors, *Proceedings of ICCCI'2009*, volume 5796 of *LNAI*, pages 437–448. Springer, 2009.

22. L.A. Nguyen and A. Szałas. An optimal tableau decision procedure for Converse-PDL. In N.-T. Nguyen, T.-D. Bui, E. Szczerbicki, and N.-B. Nguyen, editors, *Proceedings of KSE'2009*, pages 207–214. IEEE Computer Society, 2009.
23. L.A. Nguyen and A. Szałas. Checking consistency of an ABox w.r.t. global assumptions in PDL. *Fundamenta Informaticae*, 102(1):97–113, 2010.
24. L.A. Nguyen and A. Szałas. Tableaux with global caching for checking satisfiability of a knowledge base in the description logic $\mathcal{SH}$. *T. Computational Collective Intelligence*, 1:21–38, 2010.
25. L.A. Nguyen and A. Szałas. ExpTime tableau decision procedures for regular grammar logics with converse. *Studia Logica*, 98(3):387–428, 2011.
26. J.Z. Pan and I. Horrocks. Reasoning in the SHOQ(D_n) description logic. In *Proceedings of DL'2002*, volume 53 of *CEUR Workshop Proceedings*, pages 53–62, 2002.
27. V.R. Pratt. A near-optimal method for reasoning about action. *J. Comp. Syst. Sci.*, 20(2):231–254, 1980.
28. W. Rautenberg. Modal tableau calculi and interpolation. *JPL*, 12:403–423, 1983.
29. S. Tobies. *Complexity results and practical algorithms for logics in knowledge representation.* PhD thesis, RWTH-Aachen, 2001.
30. `http://owl.cs.manchester.ac.uk/navigator/`.

A Appendix: Proofs

A.1 Complexity

Let N be the size of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$. Define $\mathsf{closure}(\mathcal{R}, \mathcal{T}, \mathcal{A})$ to be the smallest set Γ of formulas such that:

1. all concepts (and subconcepts) used in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ belong to Γ,
2. if r is a role and a is an individual used in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ then $(\leq 1\, r.\{a\}) \in \Gamma$,
3. if $\forall s.C \in \Gamma$ and $r \sqsubseteq_{\mathcal{R}} s$ then $\forall r.C \in \Gamma$,
4. if $\leq 0\, s.C \in \Gamma$ then $\forall s.\overline{C} \in \Gamma$,
5. if $C \in \Gamma$ and C is not of the form $\preceq n\, r.C$ nor $\succeq n\, r.C$ then $\overline{C} \in \Gamma$,
6. if $\exists r.C \in \Gamma$ and r is a numeric role then $\succeq 1\, r.C \in \Gamma$,
7. if $\geq n\, r.C \in \Gamma$, $0 \leq m \leq \mathsf{N}$ and $m < n$ then $\succeq (n-m)\, r.C \in \Gamma$,
8. if $\leq n\, r.C \in \Gamma$, $0 \leq m \leq \mathsf{N}$ and $m \leq n$ then $\preceq (n-m)\, r.C \in \Gamma$,
9. all assertions of $\mathcal{A}$ belong to Γ,
10. if $C \in \Gamma$ and a is an individual used in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ then $a{:}C \in \Gamma$,
11. if a and b are individuals used in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ then $a \doteq b$ and $a \neq b$ belong to Γ,
12. if r is a role and a, b are individuals used in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ then $r(a,b)$ and $\neg r(a,b)$ belong to Γ.

Lemma A.1. *The number of formulas of* $\mathsf{closure}(\mathcal{R}, \mathcal{T}, \mathcal{A})$ *is of rank* $O(\mathsf{N}^3)$*, where* N *is the size of* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$*.*

Proof. The set $\Gamma = \mathsf{closure}(\mathcal{R}, \mathcal{T}, \mathcal{A})$ can be constructed by initializing Γ according to the items 1 and 9, and then repeatedly applying the rules stated in the remaining items of the list. After initialization, the set Γ has $O(\mathsf{N})$ formulas. The rules in the items 2-8 add $O(\mathsf{N}^2)$ formulas to Γ. The rule in the item 10 adds $O(\mathsf{N}^3)$ formulas to Γ (as Γ contains $O(\mathsf{N}^2)$ concepts and there are $O(\mathsf{N})$ individual names). The rules in the items 11 and 12 add $O(\mathsf{N}^3)$ formulas to Γ. Thus, at the end, Γ is of rank $O(\mathsf{N}^3)$. □

We recall below Lemma 4.6 before presenting its proof.

Lemma 4.6. *Let* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ *be a knowledge base in NNF of the logic* $\mathcal{SHOQ}$ *and let* N *be the size of* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$*. Then a* $C_{\mathcal{SHOQ}}$*-tableau for* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ *can be constructed in (at most) exponential time in* N *in the following cases:*

1. *numbers are coded in unary,*
2. *numbers are coded in binary and* $n \leq \mathsf{N}$ *for every concept* $\leq n\, r.C$ *occurring in* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$*,*
3. *numbers are coded in binary and* $n \leq \mathsf{N}$ *for every concept* $\geq n\, r.C$ *occurring in* $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$*.* □

Proof. Let us construct an arbitrary $C_{\mathcal{SHOQ}}$-tableau $G = \langle V, E, \nu \rangle$ for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$.

Let N' be the number of formulas of $\mathsf{closure}(\mathcal{R}, \mathcal{T}, \mathcal{A})$. We have $\mathsf{N}' = O(\mathsf{N}^3)$. For each $v \in V$, $\mathit{Label}(v) \subseteq \mathsf{closure}(\mathcal{R}, \mathcal{T}, \mathcal{A})$. Since nodes of G are globally cached, it follows that G has no more than $2^{\mathsf{N}'}$ nodes.

Each state has no more than $O(2^{\mathsf{N}} \cdot 2^{\mathsf{N}'} \cdot \mathsf{N})$ outgoing edges (since each outgoing edge created by the transitional full-expansion rule is characterized by a tuple $\langle X, Y, \alpha \rangle$, where X is a set of roles, Y is a set of concepts, and α is null or an individual name).[15] Thus, checking feasibility of $\mathit{ILConstraints}(v)$ for a state v is an IFDL(N, $2^{\mathsf{N}} \cdot 2^{\mathsf{N}'} \cdot \mathsf{N}$, N)-problem that satisfies the assumptions of Lemma 2.1 for the first case and satisfies the assumptions of Lemma 2.2 for the remaining two cases, and hence can be solved in (at most) exponential time in N.

Therefore, choosing a node to expand, checking whether a rule is applicable, and applying a rule can be done in (at most) exponential time in N. As each node is re-expanded at most once (by the rule (DN)), we conclude that the graph G can be constructed in (at most) exponential time in N for the considered cases. □

[15] The bound can be made tighter, e.g., using $O(\mathsf{N}^2)$ instead of N'.

A.2 Soundness

Lemma A.2. *Let $G = \langle V, E, \nu \rangle$ be a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$. Then, for every $v \in V$, $\texttt{FullLabel}(v)$ is equivalent to $Label(v)$. That is, for any interpretation $\mathcal{I}$:*

- *if v is a simple node then $(\texttt{FullLabel}(v))^{\mathcal{I}} = (Label(v))^{\mathcal{I}}$,*
- *if v is a complex node then $\mathcal{I}$ satisfies $\texttt{FullLabel}(v)$ iff it satisfies $Label(v)$.*

The proof of this lemma is straightforward.

Lemma A.3. *Let $G = \langle V, E, \nu \rangle$ be a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$. Then, for every $v \in V$, if $Type(v) =$ non-state and $w_1, \ldots, w_k$ are all the successors of v then $\texttt{FullLabel}(v)$ is satisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$ iff there exists $1 \leq i \leq k$ such that $\texttt{FullLabel}(w_i)$ is satisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$.*

The proof of this lemma is straightforward.

Let $G = \langle V, E, \nu \rangle$ be a $C_{\mathcal{SHOQ}}$-tableau. For each node v of G with $Status(v) \in \{$closed, open$\}$, let $DSTimeStamp(v)$ be the moment at which $Status(v)$ was changed to its final value (i.e., determined to be closed or open). *DSTimeStamp* stands for "determined-status time-stamp". For each node v of G with $Status(v) =$ closed-wrt(U) and for each $u \in U$, let $CSTimeStamp(v, u)$ be the first moment at which $Status(v)$ was changed to some closed-wrt(U') with $u \in U'$. *CSTimeStamp* stands for "closed-wrt-status time-stamp". For each non-state v of G, let $ETimeStamp(v)$ be the moment at which v was expanded the last time.[16]

Lemma A.4. *Let $G = \langle V, E, \nu \rangle$ be a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$. Then, for every $v \in V$:*

1. *if $Status(v) =$ closed then $\texttt{FullLabel}(v)$ is unsatisfiable w.r.t. $\mathcal{R}$ and $\mathcal{T}$,*
2. *if $Status(v) =$ closed-wrt(U) and $u \in U$ then there does not exist any model of $\langle \mathcal{R}, \mathcal{T}, \texttt{FullLabel}(u) \rangle$ that satisfies $\texttt{FullLabel}(v)$.*

Proof. We prove this lemma by induction on the above mentioned time-stamps.

Consider the first assertion of the lemma for the case when v is a simple state and $Status(v)$ is changed to closed by the subrule 2b of (UPS_3) because $ILConstraints(v)$ is infeasible. We prove the contrapositive: suppose $\mathcal{I}$ is a model of $\mathcal{R}$ and $\mathcal{T}$, and $y \in (\texttt{FullLabel}(v))^{\mathcal{I}}$; we show that $ILConstraints(v)$ is feasible. Without loss of generality, assume that $\mathcal{I}$ is finitely-branching.[17] Thus, the set $Z = \{z \in \Delta^{\mathcal{I}} \mid \langle y, z \rangle \in r^{\mathcal{I}}$ for some $r \in \mathbf{R}\}$ is finite. We compute a solution $\mathcal{S}$ for $ILConstraints(v)$ as follows.

- For each $\langle v, w \rangle \in E$ and $e \in ELabels(v, w)$ such that $\pi_T(e) =$ checkingFeasibility, set $n_{w,e} := 0$.
- For each $z \in Z$ do:
  - let $\langle w_1, e_1 \rangle, \ldots, \langle w_k, e_k \rangle$ be all the pairs such that, for each $1 \leq i \leq k$:
    - $\langle v, w_i \rangle \in E$, $e_i \in ELabels(v, w_i)$ and $\pi_T(e_i) =$ checkingFeasibility,
    - $z \in (\texttt{FullLabel}(w_i))^{\mathcal{I}}$,
    - $\langle y, z \rangle \in r^{\mathcal{I}}$ for all $r \in \pi_R(e_i)$,
    - the pair $\langle w_i, e_i \rangle$ is "maximal" in the sense that there does not exist any pair $\langle w_i', e_i' \rangle \neq \langle w_i, e_i \rangle$ such that
      - $\langle v, w_i' \rangle \in E$, $e_i' \in ELabels(v, w_i')$ and $\pi_T(e_i') =$ checkingFeasibility,
      - $z \in (\texttt{FullLabel}(w_i'))^{\mathcal{I}}$,
      - $\langle y, z \rangle \in r^{\mathcal{I}}$ for all $r \in \pi_R(e_i')$;
  - for each $1 \leq i \leq k$, set $n_{w_i,e_i} := n_{w_i,e_i} + 1$.
- $\mathcal{S} := \{x_{w,e} = n_{w,e} \mid \langle v, w \rangle \in E$, $e \in ELabels(v, w)$ and $\pi_T(e) =$ checkingFeasibility$\}$.

[16] Each non-state may be re-expanded at most once (by the rule (DN)) and each state is expanded at most once.
[17] It is known that the DL $\mathcal{SHOQ}$ has the finitely-branching model property.

We prove that $\mathcal{S}$ is a solution for $ILConstraints(v)$.

If a constraint $x_{w,e} = 0$ was added to $ILConstraints(v)$ because w got status closed then, by the inductive assumption with v replaced by w, we can conclude that $n_{w,e}$ was not increased at all and hence must be 0, which means that the constraint $x_{w,e} = 0$ is satisfied by the solution $\mathcal{S}$.

Consider a constraint $\sum\{x_{w,e} \mid \langle v,w\rangle \in E,\ e \in ELabels(v,w),\ \pi_T(e) = \mathsf{checkingFeasibility},\ r \in \pi_R(e),\ \pi_I(e) = \mathsf{null},\ D \in Label(w)\} \geq n$ of $ILConstraints(v)$ and the corresponding concept $\succeq n\, r.D$. By the assumptions about v and y, it can be derived that Z contains pairwise different $z_1, \ldots, z_n$ such that $\langle y, z_i\rangle \in r^{\mathcal{I}}$ and $z_i \in D^{\mathcal{I}}$, for $1 \leq i \leq n$. Each z_i makes $n_{w,e}$ increased by 1 for some pair $\langle w, e\rangle$ such that $\langle v,w\rangle \in E$, $e \in ELabels(v,w)$, $\pi_T(e) = \mathsf{checkingFeasibility}$, $r \in \pi_R(e)$ and $D \in \mathtt{FullLabel}(w)$. Therefore, the considered constraint is satisfied by the solution $\mathcal{S}$.

Consider a constraint $\sum\{x_{w,e} \mid \langle v,w\rangle \in E,\ e \in ELabels(v,w),\ \pi_T(e) = \mathsf{checkingFeasibility},\ r \in \pi_R(e),\ \pi_I(e) = \mathsf{null},\ D \in Label(w)\} \leq n$ of $ILConstraints(v)$ and the corresponding concept $\preceq n\, r.D$. By the assumptions about v and y, it can be derived that Z contains no more than n pairwise different elements $z_1, \ldots, z_n$ such that $\langle y, z_i\rangle \in r^{\mathcal{I}}$ and $z_i \in D^{\mathcal{I}}$, for $1 \leq i \leq n$. For each z_i, there exists at most one pair $\langle w, e\rangle$ such that $\langle v,w\rangle \in E$, $e \in ELabels(v,w)$, $\pi_T(e) = \mathsf{checkingFeasibility}$, $r \in \pi_R(e)$, $D \in \mathtt{FullLabel}(w)$ and the consideration of z_i causes $n_{w,e}$ to be increased by 1. This is due to the "maximality" of $\langle w, e\rangle$ and the nature of the transitional full-expansion rule. Therefore, the considered constraint is satisfied by the solution $\mathcal{S}$.

Now, consider the second assertion of the lemma for the case when v is a complex state and $Status(v)$ becomes $\mathsf{closed\text{-}wrt}(U)$ with $u \in U$ because of the call of $\mathtt{SetClosedWrt}(v,u)$ at the step 3b of the rule (UPS_3) due to infeasibility of the set $ILC = ILConstraints(v) \cup \{x_{w_i,e_i} = 0 \mid 1 \leq i \leq k\}$, where $\langle w_1, e_1\rangle, \ldots, \langle w_k, e_k\rangle$ are the pairs mentioned at that step of (UPS_3). We prove the contrapositive: suppose $\mathcal{I}$ is a model of $\langle \mathcal{R}, \mathcal{T}, \mathtt{FullLabel}(u)\rangle$ that satisfies $\mathtt{FullLabel}(v)$; we show that the mentioned set ILC of constraints is feasible. Without loss of generality, assume that $\mathcal{I}$ is finitely-branching. We compute a solution $\mathcal{S}$ for ILC as follows.

1. For each $\langle v,w\rangle \in E$ and $e \in ELabels(v,w)$ such that $\pi_T(e) = \mathsf{checkingFeasibility}$, set $n_{w,e} := 0$.
2. For each individual a occurring in $Label(v)$ and each $z \in \Delta^{\mathcal{I}}$ such that $\langle a^{\mathcal{I}}, z\rangle \in r^{\mathcal{I}}$ for some $r \in \mathbf{R}$ do:
   (a) let $\langle w'_1, e'_1\rangle, \ldots, \langle w'_{k'}, e'_{k'}\rangle$ be all the pairs such that, for each $1 \leq i \leq k'$:
      i. $\langle v, w'_i\rangle \in E$, $e'_i \in ELabels(v,w'_i)$, $\pi_T(e'_i) = \mathsf{checkingFeasibility}$ and $\pi_I(e'_i) = a$,
      ii. $z \in (\mathtt{FullLabel}(w'_i))^{\mathcal{I}}$,
      iii. $\langle a^{\mathcal{I}}, z\rangle \in r^{\mathcal{I}}$ for all $r \in \pi_R(e'_i)$,
      iv. the pair $\langle w'_i, e'_i\rangle$ is "maximal" in the sense that there does not exist any pair $\langle w''_i, e''_i\rangle \neq \langle w'_i, e'_i\rangle$ such that
         – $\langle v, w''_i\rangle \in E$, $e''_i \in ELabels(v,w''_i)$, $\pi_T(e''_i) = \mathsf{checkingFeasibility}$ and $\pi_I(e''_i) = a$,
         – $z \in (\mathtt{FullLabel}(w''_i))^{\mathcal{I}}$,
         – $\langle a^{\mathcal{I}}, z\rangle \in r^{\mathcal{I}}$ for all $r \in \pi_R(e''_i)$;
   (b) for each $1 \leq i \leq k'$ do
      i. if $z \neq b^{\mathcal{I}}$ for all b occurring in $Label(v)$ then $n_{w'_i,e'_i} := n_{w'_i,e'_i} + 1$;
      ii. else if $z = b^{\mathcal{I}}$ for some b occurring in $Label(v)$ and there exists $a\!:\!(\succeq l\, s.D) \in Label(v)$ such that $s \in \pi_R(e'_i)$, $D \in \mathtt{FullLabel}(w'_i)$ and $s(a,b) \notin Label(v)$ then $n_{w'_i,e'_i} := n_{w'_i,e'_i} + 1$.
3. $\mathcal{S} := \{x_{w,e} = n_{w,e} \mid \langle v,w\rangle \in E,\ e \in ELabels(v,w)$ and $\pi_T(e) = \mathsf{checkingFeasibility}\}$.

We prove that $\mathcal{S}$ is a solution for ILC.

If a constraint $x_{w,e} = 0$ was added to $ILConstraints(v)$ because w got status closed then, by the inductive assumption with v replaced by w, we can conclude that $n_{w,e}$ was not increased at all and hence must be 0, which means that the constraint $x_{w,e} = 0$ is satisfied by the solution $\mathcal{S}$.

Consider a constraint $(x_{w_i,e_i} = 0) \in ILC$ with $1 \leq i \leq k$ (the pair $\langle w_i, e_i\rangle$ was mentioned earlier). By the specification of $\langle w_i, e_i\rangle$ (at the step 3b of the rule (UPS_3)), $Status(w_i)$ is of

the form closed-wrt(U_i) with $u \in U_i$. By the inductive assumption with v replaced by w_i, we can conclude that n_{w_i,e_i} was not increased at all and hence must be 0, which means that the constraint $x_{w_i,e_i} = 0$ is satisfied by the solution $\mathcal{S}$.

Consider a concept $(a :\succeq n\, s.D) \in \mathit{Label}(v)$ and the corresponding constraint $\sum\{x_{w,e} \mid \langle v,w\rangle \in E,\ e \in \mathit{ELabels}(v,w),\ \pi_T(e) = \mathsf{checkingFeasibility},\ s \in \pi_R(e),\ \pi_I(e) = a,\ D \in \mathit{Label}(w)\} \geq n$ of $\mathit{ILConstraints}(v)$. Let $m = \sharp\{b \mid \{s(a,b), b : D\} \subseteq \mathtt{FullLabel}(v)\}$. We have that either $(a :\geq (n+m)\, s.D) \in \mathit{Label}(v)$ or $n = 1$, $m = 0$ and $(a : \exists s.D) \in \mathit{Label}(v)$. Since $\mathcal{I}$ is a model of $\mathtt{FullLabel}(v)$, there exist pairwise different $z_1, \ldots, z_{n+m}$ such that $\langle a^{\mathcal{I}}, z_i\rangle \in s^{\mathcal{I}}$ and $z_i \in D^{\mathcal{I}}$ for all $1 \leq i \leq n+m$. Note that, if $s(a,b) \in \mathit{Label}(v)$ then, by the subrule 2 of (NUS), either $b : D \in \mathtt{FullLabel}(v)$ or $b : \overline{D} \in \mathtt{FullLabel}(v)$. Since $z_i \in D^{\mathcal{I}}$ and $\mathcal{I}$ is a model of $\mathtt{FullLabel}(v)$, if $z_i = b^{\mathcal{I}}$ then $b : \overline{D} \notin \mathtt{FullLabel}(v)$. Therefore, for every $1 \leq i \leq n+m$, if $z_i = b^{\mathcal{I}}$ and $s(a,b) \in \mathit{Label}(v)$ then $b : D \in \mathtt{FullLabel}(v)$. Let $Z = \{z_1, \ldots, z_{n+m}\} - \{b^{\mathcal{I}} \mid s(a,b) \in \mathit{Label}(v)\}$. We have that $\sharp Z = n$. Each z from Z makes $n_{w,e}$ increased by 1 for some pair $\langle w,e\rangle$ such that $\langle v,w\rangle \in E$, $e \in \mathit{ELabels}(v,w)$, $\pi_T(e) = \mathsf{checkingFeasibility}$, $\pi_I(e) = a$, $s \in \pi_R(e)$ and $D \in \mathit{Label}(w)$. It follows that the considered constraint is satisfied by the solution $\mathcal{S}$.

Consider a concept $(a :\preceq n\, r.C) \in \mathit{Label}(v)$ and the corresponding constraint $\sum\{x_{w,e} \mid \langle v,w\rangle \in E,\ e \in \mathit{ELabels}(v,w),\ \pi_T(e) = \mathsf{checkingFeasibility},\ r \in \pi_R(e),\ \pi_I(e) = a,\ C \in \mathit{Label}(w)\} \leq n$ of $\mathit{ILConstraints}(v)$. Let $m = \sharp\{b \mid \{r(a,b), b : C\} \subseteq \mathtt{FullLabel}(v)\}$. We have that $(a :\leq (n+m)\, r.C) \in \mathit{Label}(v)$. Since $\mathcal{I}$ is a model of $\mathtt{FullLabel}(v)$, it follows that $a^{\mathcal{I}} \in (\leq (n+m)\, r.C)^{\mathcal{I}}$. Let $Z_1 = \{b^{\mathcal{I}} \mid \{r(a,b), b : C\} \subseteq \mathtt{FullLabel}(v)\}$. Due to the subrule 3 of (NUS), we have that $\sharp Z_1 = m$.

Note that if $\langle v,w\rangle \in E$, $e \in \mathit{ELabels}(v,w)$, $\pi_T(e) = \mathsf{checkingFeasibility}$, $\pi_I(e) = a$, $r \in \pi_R(e)$ and $C \in \mathtt{FullLabel}(w)$ then $n_{w,e}$ is increased only due to some z such that $\langle a^{\mathcal{I}}, z\rangle \in r^{\mathcal{I}}$ and $z \in C^{\mathcal{I}}$. Due to the "maximality" of $\langle w,e\rangle$ and the nature of the transitional full-expansion rule, for such a z there exists at most one pair $\langle v,w\rangle$ such that $\langle v,w\rangle \in E$, $e \in \mathit{ELabels}(v,w)$, $\pi_T(e) = \mathsf{checkingFeasibility}$, $\pi_I(e) = a$, $r \in \pi_R(e)$, $C \in \mathit{Label}(w)$ and the consideration of z causes $n_{w,e}$ to be increased by 1. Since $a^{\mathcal{I}} \in (\leq (n+m)\, r.C)^{\mathcal{I}}$, to prove that the considered constraint is satisfied by the solution $\mathcal{S}$, it suffices to show that if $z \in Z_1$ causes $n_{w'_i,e'_i}$ to be increased by 1 at the step 2(b)ii then $r \notin \pi_R(e'_i)$ or $C \notin \mathit{Label}(w'_i)$. Suppose the contrary. We have that:

- $\{a :\leq (n+m)\, r.C, r(a,b), b : C\} \subseteq \mathtt{FullLabel}(v)$, (1)
- $\langle v, w'_i\rangle \in E,\ e'_i \in \mathit{ELabels}(v, w'_i),\ \pi_T(e'_i) = \mathsf{checkingFeasibility}$ and $\pi_I(e'_i) = a$, (2)
- $b^{\mathcal{I}} \in (\mathit{Label}(w'_i))^{\mathcal{I}}$ and $\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in (r')^{\mathcal{I}}$ for all $r' \in \pi_R(e'_i)$, (3)
- $a : (\succeq l\, s.D) \in \mathit{Label}(v),\ s \in \pi_R(e'_i),\ D \in \mathit{Label}(w'_i)$ and $s(a,b) \notin \mathit{Label}(v)$, (4)
- $r \in \pi_R(e'_i)$ and $C \in \mathit{Label}(w'_i)$. (5)

Since both s and r belong to $\pi_R(e'_i)$ (by (4) and (5)), there exist roles

$$r_0 = r, r_1, \ldots, r_{h-1}, r_h = s \text{ and } s_1, \ldots, s_h, \text{ all belonging to } \pi_R(e'_i) \tag{6}$$

such that, for every $1 \leq j \leq h$:

- $s_j \sqsubseteq_{\mathcal{R}} r_{j-1}$ and $s_j \sqsubseteq_{\mathcal{R}} r_j$, (7)
- $\mathit{Label}(v)$ contains $a : \exists s_j.D'_j$ or $a :\geq n_j\, s_j.D'_j$ for some $D'_j \in \mathit{Label}(w'_i)$ and $n_j > 0$, (8)
- if $j < h$ then $\mathit{Label}(v)$ contains $a :\leq m_j\, r_j.C'_j$ for some $C'_j \in \mathit{Label}(w'_i)$ and m_j. (9)

Note that the subrule 4 of (NUS) was not applicable to v. Having (1), (6), (7) and (8), we derive that $s_1(a,b) \in \mathit{Label}(v)$ or $\neg s_1(a,b) \in \mathit{Label}(v)$. Since (6) and (3), $\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in s_1^{\mathcal{I}}$. Since $\mathcal{I}$ is a model of $\mathtt{FullLabel}(v)$, it follows that $\neg s_1(a,b) \notin \mathit{Label}(v)$, and hence $s_1(a,b) \in \mathit{Label}(v)$. Since $s_1 \sqsubseteq r_1$ (by (7)), by the rule (US_1), we also have that $r_1(a,b) \in \mathit{Label}(v)$. Analogously, using

also (9), for every j from 1 to h, we can derive that $s_j(a,b) \in \mathit{Label}(v)$ and $r_j(a,b) \in \mathit{Label}(v)$. Since $s = r_h$, it follows that $s(a,b) \in \mathit{Label}(v)$, which contradicts (4). This completes the induction step for the second assertion of the lemma for the case when v is a complex state and $\mathit{Status}(v)$ becomes $\mathsf{closed\text{-}wrt}(U)$ with $u \in U$ because of the call of $\mathtt{SetClosedWrt}(v,u)$ at the step 3b of the rule (UPS_3) due to infeasibility of ILC.

The induction steps for:

- the first assertion of the lemma for the case when v is a complex state and $\mathit{Status}(v)$ is changed to closed by the subrule 2b of (UPS_3) because $\mathit{ILConstraints}(v)$ is infeasible,
- the second assertion of the lemma for the case when v is a simple state and $\mathit{Status}(v)$ becomes $\mathsf{closed\text{-}wrt}(U)$ with $u \in U$ because of the call of $\mathtt{SetClosedWrt}(v,u)$ at the step 3b of the rule (UPS_3) due to infeasibility of the corresponding set of constraints

can be proved in a similar way as done above for the two dual cases.

The induction steps for the other cases (that correspond to the subrule 1 of (UPS_1), the rule (UPS_2) and the subrules 1b, 1c, 2a, 3a of (UPS_3)) are straightforward. □

Corollary A.5 (Soundness of $C_{\mathcal{SHOQ}}$). *If $G = \langle V, E, \nu\rangle$ is a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ and $\mathit{Status}(\nu) = \mathsf{closed}$ then $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ is unsatisfiable.*

This corollary directly follows from Lemma A.4.

A.3 Completeness

We prove completeness of $C_{\mathcal{SHOQ}}$ via model graphs. The technique has been used for other logics (e.g., in [28, 4, 15, 1]). A *model graph* is a tuple $\langle \Delta, \mathbb{I}, \mathbb{C}, \mathbb{E}\rangle$, where:

- Δ is a non-empty and finite set,
- $\mathbb{I}$ is a mapping that associates each individual name with an element of Δ,
- $\mathbb{C}$ is a mapping that associates each element of Δ with a set of concepts,
- $\mathbb{E}$ is a mapping that associates each role with a binary relation on Δ.

A model graph $\langle \Delta, \mathbb{I}, \mathbb{C}, \mathbb{E}\rangle$ is *consistent* and *$\mathcal{R}$-saturated* if every $x \in \Delta$ satisfies:[18]

- $\mathbb{C}(x)$ does not contain $\bot$ nor any pair $C, \overline{C}$ (10)
- if $\langle x,y\rangle \in \mathbb{E}(r)$ and $r \sqsubseteq_{\mathcal{R}} s$ then $\langle x,y\rangle \in \mathbb{E}(s)$ (11)
- if $\{a\} \in \mathbb{C}(x)$ then $\mathbb{I}(a) = x$ (12)
- if $C \sqcap D \in \mathbb{C}(x)$ then $\{C, D\} \subseteq \mathbb{C}(x)$ (13)
- if $C \sqcup D \in \mathbb{C}(x)$ then $C \in \mathbb{C}(x)$ or $D \in \mathbb{C}(x)$ (14)
- if $\forall s.C \in \mathbb{C}(x)$ and $r \sqsubseteq_{\mathcal{R}} s$ then $\forall r.C \in \mathbb{C}(x)$ (15)
- if $\langle x,y\rangle \in \mathbb{E}(r)$ and $\forall r.C \in \mathbb{C}(x)$ then $C \in \mathbb{C}(y)$ (16)
- if $\langle x,y\rangle \in \mathbb{E}(r)$, $\mathit{trans}_{\mathcal{R}}(r)$ and $\forall r.C \in \mathbb{C}(x)$ then $\forall r.C \in \mathbb{C}(y)$ (17)
- if $\exists r.C \in \mathbb{C}(x)$ then $\exists y \in \Delta$ such that $\langle x,y\rangle \in \mathbb{E}(r)$ and $C \in \mathbb{C}(y)$ (18)
- if $(\geq n\, r.C) \in \mathbb{C}(x)$ then $\sharp\{\langle x,y\rangle \in \mathbb{E}(r) \mid C \in \mathbb{C}(y)\} \geq n$ (19)
- if $(\leq n\, r.C) \in \mathbb{C}(x)$ then $\sharp\{\langle x,y\rangle \in \mathbb{E}(r) \mid C \in \mathbb{C}(y)\} \leq n$ (20)
- if $(\leq n\, r.C) \in \mathbb{C}(x)$ and $\langle x,y\rangle \in \mathbb{E}(r)$ then $C \in \mathbb{C}(y)$ or $\overline{C} \in \mathbb{C}(y)$. (21)

Given a model graph $M = \langle \Delta, \mathbb{I}, \mathbb{C}, \mathbb{E}\rangle$, the *$\mathcal{R}$-model corresponding to M* is the interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}}\rangle$ where:

- $a^{\mathcal{I}} = \mathbb{I}(a)$ for every individual name a,

[18] A consistent and $\mathcal{R}$-saturated model graph is like a Hintikka structure.

- $A^{\mathcal{I}} = \{x \in \Delta \mid A \in \mathbb{C}(x)\}$ for every concept name A,
- $r^{\mathcal{I}} = \mathbb{E}'(r)$ for every role name $r \in \mathbf{R}$, where $\mathbb{E}'(r)$ for $r \in \mathbf{R}$ are the smallest binary relations on Δ such that:
  - $\mathbb{E}(r) \subseteq \mathbb{E}'(r)$,
  - if $r \sqsubseteq_{\mathcal{R}} s$ then $\mathbb{E}'(r) \subseteq \mathbb{E}'(s)$,
  - if $\mathit{trans}_{\mathcal{R}}(r)$ then $\mathbb{E}'(r) \circ \mathbb{E}'(r) \subseteq \mathbb{E}'(r)$.

Note that the smallest binary relations mentioned above always exist: for each $r \in \mathbf{R}$, initialize $\mathbb{E}'(r)$ with $\mathbb{E}(r)$; then, while one of the above mentioned condition is not satisfied, extend the corresponding $\mathbb{E}'(r)$ minimally to satisfy the condition.

Lemma A.6. *If $\mathcal{I}$ is the $\mathcal{R}$-model corresponding to a consistent $\mathcal{R}$-saturated model graph $\langle \Delta, \mathbb{I}, \mathbb{C}, \mathbb{E} \rangle$, then $\mathcal{I}$ is a model of $\mathcal{R}$ and, for every $x \in \Delta$ and $C \in \mathbb{C}(x)$, we have that $x \in C^{\mathcal{I}}$.*

The first assertion of this lemma clearly holds. The second assertion can be proved by induction on the structure of C in a straightforward way.

Let $G = \langle V, E, \nu \rangle$ be a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$.

Let $v \in V$ be a complex non-state with $\mathit{Status}(v) \neq \mathsf{closed}$. A *saturation path* of v is a sequence $v_0 = v, v_1, \ldots, v_k$ of nodes of G, with $k \geq 1$, such that $\mathit{Type}(v_k) = \mathsf{state}$ and

- for every $1 \leq i \leq k$, $\mathit{Status}(v_i) \neq \mathsf{closed}$
- for every $0 \leq i < k$, $\mathit{Type}(v_i) = \mathsf{non\text{-}state}$ and $\langle v_i, v_{i+1} \rangle \in E$.

Observe that each saturation path of v is finite.[19] Furthermore, if v_i is a non-state with $\mathit{Status}(v_i) \neq \mathsf{closed}$ then v_i has a successor v_{i+1} with $\mathit{Status}(v_{i+1}) \neq \mathsf{closed}$. Therefore, v has at least one saturation path.

Let $u \in V$ be a complex state and $v \in V$ be a simple non-state such that $\mathit{Status}(u) \neq \mathsf{closed}$, $\mathit{Status}(v) \neq \mathsf{closed}$, $\mathit{Status}(v) \neq \mathsf{closed\text{-}wrt}(\{u, \ldots\})$ and v may affect the status of the root ν via a path through u. A *saturation path of v w.r.t. u* is a sequence $v_0 = v, v_1, \ldots, v_k$ of nodes of G, with $k \geq 1$, such that either $\mathit{Type}(v_k) = \mathsf{state}$ or $\mathit{Status}(v_k) = \mathsf{blocked}$, and

- for every $1 \leq i \leq k$, $\mathit{Status}(v_i)$ is not closed nor $\mathsf{closed\text{-}wrt}(\{u, \ldots\})$,
- for every $0 \leq i < k$, $\mathit{Type}(v_i) = \mathsf{non\text{-}state}$ and $\langle v_i, v_{i+1} \rangle \in E$.

Observe that each saturation path of v w.r.t. u is finite (see the footnote 19). Furthermore, if v_i is a non-state with $\mathit{Status}(v_i)$ different from closed and $\mathsf{closed\text{-}wrt}(\{u, \ldots\})$, then v_i has a successor v_{i+1} with $\mathit{Status}(v_{i+1})$ different from closed and $\mathsf{closed\text{-}wrt}(\{u, \ldots\})$. Therefore, v has at least one saturation path w.r.t. u.

Lemma A.7 (Completeness of $C_{\mathcal{SHOQ}}$). *Let $G = \langle V, E, \nu \rangle$ be a $C_{\mathcal{SHOQ}}$-tableau for $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$. Suppose $\mathit{Status}(\nu) \neq \mathsf{closed}$. Then $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ is satisfiable.*

Proof. The root ν has a saturation path $u_0, \ldots, u_k$ with $u_0 = \nu$. Let $u = u_k$. We define a model graph $M = \langle \Delta, \mathbb{I}, \mathbb{C}, \mathbb{E} \rangle$ as follows:

1. Let Δ_0 be the set of all individual names a such that $\mathit{IndRepl}(u)(a) = a$. Set $\Delta := \Delta_0$ and $\mathbb{I} := \mathit{IndRepl}(u)$. If $a \in \mathbf{I}$ does not occur in $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ then define $\mathbb{I}(a)$ to be some individual occurring in Δ_0. For each $a \in \Delta_0$, mark a as *unresolved*[20] and set $\mathbb{C}(a) := \{C \mid (a : C) \in \mathtt{FullLabel}(u)\}$. For each role r, set $\mathbb{E}(r) := \{\langle a, b \rangle \mid r(a, b) \in \mathtt{FullLabel}(u)\}$.
2. For every *unresolved* node $y \in \Delta$ do:
   (a) If $y \in \Delta_0$ then let $v = u$ and
   $WE = \{\langle w, e \rangle \mid \langle v, w \rangle \in E, e \in \mathit{ELabels}(v, w) \text{ and } \pi_I(e) = y\}$.

[19] If a non-state v_{i+1} is a successor of a non-state v_i then either $\sharp\{a \in \mathbf{I} \mid \mathit{IndRepl}(v_{i+1})(a) = a\} < \sharp\{a \in \mathbf{I} \mid \mathit{IndRepl}(v_i)(a) = a\}$ or $\sharp\{a \in \mathbf{I} \mid \mathit{IndRepl}(v_{i+1})(a) = a\} = \sharp\{a \in \mathbf{I} \mid \mathit{IndRepl}(v_i)(a) = a\}$ and $\mathtt{FullLabel}(v_{i+1}) \supset \mathtt{FullLabel}(v_i)$. Recall also that $\mathtt{FullLabel}(v_{i+1})$ is a subset of $\mathsf{closure}(\mathcal{R}, \mathcal{T}, \mathcal{A})$.

[20] Each node of M will be marked either as *unresolved* or as *resolved*.

(b) Else:
  i. Let $v = f(y)$.
  ($*$ f is a constructed mapping that associates each node of M not belonging to Δ_0 with a simple state of G; as a maintained property of f, $\mathit{Status}(v) \neq \mathsf{closed}$, $\mathit{Status}(v) \neq \mathsf{closed\text{-}wrt}(\{u, \ldots\})$ and $\mathbb{C}(y)$ is the set obtained from $\mathtt{FullLabel}(v)$ by replacing every individual b by $\mathit{IndRepl}(u)(b)$ when $\mathit{IndRepl}(u)(b)$ is defined. $*$)
  ii. Let $WE = \{\langle w, e\rangle \mid \langle v, w\rangle \in E, e \in \mathit{ELabels}(v, w)\}$.
(c) Let $\mathit{ILC} = \mathit{ILConstraints}(v) \cup \{x_{w,e} = 0 \mid \langle w, e\rangle \in WE,\ \pi_T(e) = \mathsf{checkingFeasibility},\ \mathit{Status}(w) = \mathsf{closed\text{-}wrt}(\{u, \ldots\})\}$.
($*$ ILC is feasible because $\mathit{Status}(v) \neq \mathsf{closed}$ and $\mathit{Status}(v) \neq \mathsf{closed\text{-}wrt}(\{u, \ldots\})$. $*$)
(d) Fix a solution of ILC, and for each $\langle w, e\rangle \in WE$:
  i. if $\pi_T(e) = \mathsf{testingClosedness}$ then let $n_{w,e} = 1$,
  ii. else let $n_{w,e}$ be the value of $x_{w,e}$ in that solution.
(e) Delete from WE all the pairs $\langle w, e\rangle$ with $n_{w,e} = 0$.
(f) For each $\langle w_0, e\rangle \in WE$ do:
  i. Let $w_0, \ldots, w_h$ be a saturation path of w_0 w.r.t. u.
  ii. Let X be the set obtained from $\mathtt{FullLabel}(w_h)$ by replacing every individual b by $\mathit{IndRepl}(u)(b)$ when $\mathit{IndRepl}(u)(b)$ is defined.
  iii. If $\mathit{Status}(w_h) = \mathsf{blocked}$ then:
    ($*$ Observe that $n_{w_0,e}$ must be 1 due to the rule (US_3) and the construction of $\mathit{ILConstraints}(v)$ by the transitional full-expansion rule. $*$)
    A. Let $\{a\}$ be an element of X.
    ($*$ Observe that, due to the specification of X, $\mathit{IndRepl}(u)(a) = a$ and $a \in \Delta_0$. $*$)
    B. For each $r \in \pi_R(e)$, add $\langle y, a\rangle$ to $\mathbb{E}(r)$.
  iv. Else, for $i := 1$ to $n_{w_0,e}$ do:
    A. Add a new element z to Δ and mark z as *unresolved*.
    B. For each $r \in \pi_R(e)$, add $\langle y, z\rangle$ to $\mathbb{E}(r)$.
    C. Set $\mathbb{C}(z) := X$ and $f(z) := w_h$.
    ($*$ Observe that the mentioned properties of f are maintained here. $*$)
(g) Mark y as *resolved*.

The defined model graph M may be infinite. It consists of a finite base created at the step 1 and disjoint trees created at the step 2 possibly with edges coming back directly to Δ_0 (nodes of the base) due to nominals.

It is straightforward to prove that M is a consistent $\mathcal{R}$-saturated model graph.

Observe that:

- For any individual name b, if $a = \mathit{IndRepl}(u)(b)$ then $\mathit{IndRepl}(u)(a) = a$ and $\mathbb{I}(a) = a$.
- If $(a : C) \in \mathcal{A}$ then the concept obtained from C by replacing every individual b by $\mathit{IndRepl}(u)(b)$ (when $\mathit{IndRepl}(u)(b)$ is defined) belongs to $\mathbb{C}(a')$, where $a' = \mathit{IndRepl}(u)(a)$.
- If $r(a, b) \in \mathcal{A}$ then $\langle a', b'\rangle \in \mathbb{E}(r)$, where $a' = \mathit{IndRepl}(u)(a)$ and $b' = \mathit{IndRepl}(u)(b)$.
- If $a \not\approx b \in \mathcal{A}$ then $a' \not\approx b' \in \mathit{Label}(u)$, where $a' = \mathit{IndRepl}(u)(a)$ and $b' = \mathit{IndRepl}(u)(b)$. Since $\mathit{Status}(u) \neq \mathsf{closed}$, we have that $a' \neq b'$.
- For every $C \in \mathcal{T}$, the concept obtained from C by replacing every individual b by $\mathit{IndRepl}(u)(b)$ (when $\mathit{IndRepl}(u)(b)$ is defined) belongs to $\mathbb{C}(x)$ for all $x \in \Delta$.

Hence, by Lemma A.6, the interpretation corresponding to M is a model of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$. □