

# On the Decidability of Expressive Description Logics with Transitive Closure and Regular Role Expressions

Jean Christoph Jung<sup>1</sup>, Carsten Lutz<sup>1</sup>, Thomas Zeume<sup>2</sup>

<sup>1</sup>University of Bremen, Germany

<sup>2</sup>Ruhr University Bochum, Germany

{jeanjung,clu}@uni-bremen.de, thomas.zeume@rub.de

## Abstract

We consider fragments of the description logic  $SHOIF$  extended with regular expressions on roles. Our main result is that satisfiability and finite satisfiability are decidable in two fragments  $SHOIF_{reg}^1$  and  $SHOIF_{reg}^2$ , NEXPTIME-complete for the former and in 2NEXPTIME for the more expressive latter fragment. Both fragments impose restrictions on regular role expressions of the form  $\alpha^*$ .  $SHOIF_{reg}^1$  encompasses the extension of  $SHOIF$  with transitive closure of roles (when functional roles have no subroles) and the modal logic of linear orders and successor, with converse. Consequently, these logics are also decidable and NEXPTIME-complete.

## 1 Introduction

There has been a long quest for description logics (DLs) that are as expressive as possible while still being decidable, culminating in the standardization of the OWL 2 DL ontology language by the W3C (OWL Working Group 2009). In its essence, OWL 2 DL is identical to a very expressive DL called  $SROIQ$  (Horrocks, Kutz, and Sattler 2006). An important feature of  $SROIQ$  and related DLs such as  $SHOIF$  and  $ALCOIF$  is that they support a certain combination of expressive means, namely nominals, inverse roles, and counting on roles. It is well-known that this combination causes computational challenges and results in an intricate model theory. For example, satisfiability is NEXPTIME-complete in  $ALCOIF$  while it becomes EXPTIME-complete and conceptually much simpler if any of the mentioned expressive means is dropped. As another example, consider the extension of  $ALCOIF$  with fixed points, a notational variant of the  $\mu$ -calculus extended with nominals, converse, and functional relations. Satisfiability in this logic is undecidable (Bonatti and Peron 2004) while it becomes decidable when any of the three expressive means is dropped (Sattler and Vardi 2001; Kupferman, Sattler, and Vardi 2002; Bonatti et al. 2008).

A traditional feature of DLs not available in OWL 2 is regular expressions on roles (Baader 1991; Baader et al. 2017; Calvanese, Eiter, and Ortiz 2009). Using these, we can for example describe a person of royal descent as

Person  $\sqcap \exists(\text{hasFather} + \text{hasMother})^+ . (\text{King} \sqcup \text{Queen})$

where ‘+’ denotes role union and ‘.’ transitive closure of a role. The extension  $ALCOIF_{reg}$  of the aforementioned

$ALCOIF$  with such expressions is a notational variant of propositional dynamic logic (PDL) extended with nominals, converse, and functional relations. It is a long standing open problem whether satisfiability in this logic is decidable and the same is true for the more expressive OWL 2 DL extended with regular expressions on roles. It is known, however, that modest extensions of this version of PDL are undecidable. One of them is the extended  $\mu$ -calculus discussed above. Another one is the extension with  $\omega$ -regular expressions on roles that speak about infinite paths rather than about finite ones like ordinary regular expressions (Rudolph 2016). From a DL perspective, arguably the most important and useful part of regular expressions on roles is transitive closure. The decidability status of very expressive DLs with transitive closure has so far been somewhat unclear. It has been stated in (Le Duc, Lamolle, and Curé 2013) that satisfiability is decidable in the extension of  $SHOIQ$  with transitive closure, but there are problems in the construction and a corrected version has not yet been published (Le Duc 2019).

We study fragments of the description logic  $SHOIF_{reg}$ , that is  $SHOIF$  extended with regular expressions on roles: role composition, role union, transitive closure of roles, and a PDL-like test operator. We generally assume that functionality restrictions are declared globally in the TBox. Our main contribution is to identify two non-trivial fragments in which satisfiability is decidable: in  $SHOIF_{reg}^1$ , every regular (sub)role expression of the form  $\alpha^*$  contains either no functional role or only a single role name (and possibly its inverse). In the more expressive  $SHOIF_{reg}^2$ , every regular (sub)role expression  $\alpha^*$  must define a regular language  $L(\alpha^*)$  such that no  $w \in L(\alpha^*)$  contains an infix of the form  $R\beta S$  with  $R$  and  $S$  distinct functional roles (or their inverse) and  $\beta$  a composition of tests. In addition, both fragments disallow subroles of functional roles. We show that satisfiability in  $SHOIF_{reg}^1$  is decidable and NEXPTIME-complete and thus not harder than in  $SHOIF$  without regular expressions, and that satisfiability in  $SHOIF_{reg}^2$  is decidable in 2NEXPTIME. We establish the same results for finite satisfiability which does not coincide with the unrestricted case. Note that  $SHOIF_{reg}^1$  contains the extension  $SHOIF^+$  of  $SHOIF$  with transitive closure of roles and thus our results imply that satisfiability and finite satisfiability in  $SHOIF^+$  are also decidable and NEXPTIME-complete (when functional roles have no subroles).

To obtain our upper bounds, we first observe that role hierarchies can be removed from  $SHOIF_{reg}^1$  and  $SHOIF_{reg}^2$  by a polynomial time reduction. We then introduce a suitable tree automata model, show that its non-emptiness problem is decidable in NEXPTIME both on finite and infinite trees, and then (essentially) encode our satisfiability problems as a non-emptiness problem. The mentioned automata are two-way alternating Büchi tree automata with existential Presburger constraints on the Parikh image (2ABTA $_{\exists PP}$ ). They are related to Parikh automata as originating from (Klaedtke and Rueß 2003), but use the constraints in a more restrictive way—instead of maintaining the constraints throughout a run, our automata correspond to the intersection of a traditional tree automaton with existential Presburger constraints over multiplicities of alphabet symbols in the accepted tree. Our decision procedure for non-emptiness relies on a translation of context-free grammars into existential Presburger formulas by Verma, Seidl, and Schwentick (2005); getting a NEXPTIME upper bound involves a subtle analysis in the infinite case. The encodings of satisfiability into non-emptiness are based on decompositions of the TBox into parts that are related only loosely in terms of multiplicities of types.

Another interesting perspective on our results emerges from putting them into the perspective of GC<sup>2</sup>, the guarded two-variable fragment of first-order logic with counting quantifiers (Pratt-Hartmann 2007). GC<sup>2</sup> easily becomes undecidable when relations can be declared to have special semantic properties such as being a linear order (Kieronski 2011) or an equivalence relation (Pratt-Hartmann 2015). In contrast, our results imply that this is not the case in  $ALCOIF$ , a very significant fragment of GC<sup>2</sup>. In fact, we can express in  $ALCOIF_{reg}$  that a role is an equivalence relation or a linear order, using fresh helper symbols in the latter case. Consequently,  $ALCOIF$  extended with these semantic properties (even simultaneously) is still decidable and NEXPTIME-complete. While decidability results for GC<sup>2</sup> and its non-guarded extension C<sup>2</sup> often rely on restricting the number of relations with semantic properties (Pratt-Hartmann 2015; Schwentick and Zeume 2012; Zeume and Harwath 2016), we impose no such restriction. Another interesting semantic property is that a relation is a forest. It has been shown in (Kotek et al. 2015) that finite satisfiability in  $ALCOIQ$  extended with forests is decidable. Our results capture the  $ALCOIF$  fragment of this DL, reprove decidability in NEXPTIME of finite satisfiability and establish the same upper bound for unrestricted satisfiability.

Proof details are deferred to the appendix of the long version at <http://www.informatik.uni-bremen.de/tdki/research/>.

## 2 Preliminaries

Let  $N_C$ ,  $N_R$ , and  $N_I$  be countably infinite sets of *concept*, *role*, and *individual names*.  $ALCOIF_{reg}$ -concepts  $C$ , roles  $R$ , and regular roles  $\alpha$  are defined by the grammar

$$\begin{aligned} C &::= A \mid \{a\} \mid \neg C \mid C \sqcap C \mid \exists \alpha. C \\ R &::= r \mid r^- \\ \alpha &::= C? \mid R \mid \alpha^* \mid \alpha \cdot \alpha \mid \alpha + \alpha \end{aligned}$$

where  $A$  ranges over  $N_C$ ,  $a$  over  $N_I$ , and  $r$  over role names from  $N_R$ . A role of the form  $r^-$  is called an *inverse role*. We write  $\perp$  as an abbreviation for  $A \sqcap \neg A$  with  $A$  a fixed concept name,  $\top$  for  $\neg \perp$ ,  $C \sqcup D$  for  $\neg(\neg C \sqcap \neg D)$ ,  $\forall \alpha. C$  for  $\neg \exists \alpha. \neg C$ , and  $r^+$  for  $r \cdot r^*$ . We refer to a concept of the form  $\{a\}$  as a *nominal* and to a role of the form  $C?$  as a *test*. For brevity, we often drop the role composition operator ‘ $\cdot$ ’, writing e.g.  $rr$  instead of  $r \cdot r$ . The restriction that the inverse operator  $\cdot^-$  is applied only to role names  $r$  is without loss of generality as this operator can be ‘pushed down’ over all other role operators and dropped when applied to tests.

A  $SHOIF_{reg}$ -TBox  $\mathcal{T}$  is a finite set of *concept inclusions* (CIs)  $C \sqsubseteq D$  with  $C, D$   $ALCOIF_{reg}$ -concepts, *role inclusions* (RIs)  $R \sqsubseteq S$  with  $R, S$  roles, and *functionality assertions*  $\text{func}(R)$  with  $R$  a role. We use  $\text{Fn}(\mathcal{T})$  to denote the set of role names  $r$  such that  $\mathcal{T}$  contains  $\text{func}(r)$  or  $\text{func}(r^-)$ .  $SHOIF$ -TBoxes typically also admit transitivity assertions  $\text{trans}(r)$ ,  $r$  a role name. This is syntactic sugar in  $SHOIF_{reg}$  since we can simulate a transitive role  $r$  by the role  $r^+$ . We do not admit regular roles in role inclusions as this is known to cause undecidability (Le Duc and Lamolle 2010).

An  $ALCOIF_{reg}$ -TBox is a  $SHOIF_{reg}$ -TBox that does not contain role inclusions. The letter  $\mathcal{F}$  and subscript  $_{reg}$  in  $ALCOIF_{reg}$  and in  $SHOIF_{reg}$  indicate the presence of functionality assertions and of regular roles. It should thus be clear what we mean, for example, with  $ALCOIF_{reg}$ -concepts and with  $ALCOIF$ -TBoxes. We use  $SHOIF^+$  to denote the extension of  $SHOIF$  with transitive closure of roles, that is, the fragment of  $SHOIF_{reg}$  in which all regular roles must be of the form  $R$  or  $R^+$  with  $R$  a role name or its inverse. For a syntactic object  $O$  such as a TBox and a concept, we write  $\|O\|$  to denote the *size* of  $O$ , that is, the number of symbols needed to write  $O$  with each concept, role, and individual name contributing one symbol.

The semantics is defined in terms of *interpretations*  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$  where  $\Delta^{\mathcal{I}}$  is a non-empty set called the *domain* and  $\cdot^{\mathcal{I}}$  is the *interpretation function* that assigns to every concept name  $A$  a subset  $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ , to every role name  $r$  a subset  $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ , and to every individual name  $a$  an element  $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ . We can extend  $\mathcal{I}$  to compound concepts and roles in the standard way, see (Baader et al. 2017) for details. An interpretation  $\mathcal{I}$  *satisfies* a CI  $C \sqsubseteq D$  if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ , an RI  $R \sqsubseteq S$  if  $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ , and a functionality assertion  $\text{func}(R)$  if  $R^{\mathcal{I}}$  is a partial function. An interpretation is a *model* of a TBox if it satisfies all inclusions and assertions in it. A TBox is (*finitely*) *satisfiable* if it has a (finite) model.

With (*finite*) *satisfiability in a DL*  $\mathcal{L}$ , we mean the problem to decide whether a given  $\mathcal{L}$ -TBox has a (finite) model. It is well-known that  $ALCOIF$  does not have the finite model property, that is, finite and unrestricted satisfiability do not coincide. For example, the  $ALCOIF$ -TBox

$$\top \sqsubseteq \exists s. A, \quad A \sqsubseteq \forall r^- . \perp, \quad \top \sqsubseteq \exists r. \top, \quad \text{func}(r^-)$$

is satisfiable, but not finitely satisfiable. In all DLs considered in this paper, other standard versions of satisfiability such as satisfiability of a concept w.r.t. a TBox or of an ABox w.r.t. a TBox (Baader et al. 2017) can be reduced in polynomial time to TBox satisfiability as defined above.

$SHOIF_{reg}$  is a fragment of  $\mathcal{L}_{\omega_1\omega}$ , the extension of first-order logic with countably infinite conjunctions and disjunctions. Since  $\mathcal{L}_{\omega_1\omega}$  has the downward Löwenheim-Skolem property (Keisler 1971), we obtain the following.

**Proposition 1** *Every satisfiable  $SHOIF_{reg}$ -TBox is satisfiable in a model with countable domain.*

It is thus w.l.o.g. to assume that domains of interpretations are countable. We shall do so from now on.

### 3 Fragments and Basic Observations

We define the fragments  $SHOIF_{reg}^1$  and  $SHOIF_{reg}^2$ . A regular role  $\alpha$  can be viewed as a regular expression over the alphabet that consists of all roles and tests in  $\alpha$ . We use  $L(\alpha)$  to denote the regular language defined by  $\alpha$ .

**Definition 2** *A  $SHOIF_{reg}^1$ -TBox  $\mathcal{T}$  is a  $SHOIF_{reg}$ -TBox such that if a regular role  $\alpha^*$  occurs in  $\mathcal{T}$ , then*

(\*<sub>1</sub>) *if a role name  $r \in \text{Fn}(\mathcal{T})$  occurs in  $\alpha$  outside of tests, then no other role name occurs in  $\alpha$  outside of tests.*

*A  $SHOIF_{reg}^2$ -TBox  $\mathcal{T}$  is a  $SHOIF_{reg}$ -TBox such that if a regular role  $\alpha^*$  occurs in  $\mathcal{T}$ , then*

(\*<sub>2</sub>) *no  $w \in L(\alpha^*)$  contains an infix of the form  $R\beta S$  with  $R \in \{r, r^-\}$ ,  $S \in \{s, s^-\}$ ,  $r, s \in \text{Fn}(\mathcal{T})$ ,  $r \neq s$ , and  $\beta$  a composition of tests.*

Moreover, for both  $SHOIF_{reg}^1$ - and  $SHOIF_{reg}^2$ -TBoxes, we require that functional roles have no subroles, that is, if  $\mathcal{T}$  contains  $S \sqsubseteq r$  or  $S \sqsubseteq r^-$ , then  $r \notin \text{Fn}(\mathcal{T})$ .

We also define corresponding fragments  $ALCCOIF_{reg}^1$  and  $ALCCOIF_{reg}^2$  of  $ALCCOIF_{reg}$  in the expected way. Clearly,  $SHOIF_{reg}^1$  is a fragment of  $SHOIF_{reg}^2$  and  $SHOIF^+$  is a fragment of  $SHOIF_{reg}^1$  subject to the condition that functional roles have no subroles. It can be shown that  $SHOIF_{reg}^2$  is more expressive than  $SHOIF_{reg}^1$  using the fact that it makes available additional regular roles such as  $(r+s)^*$  with  $r \in \text{Fn}(\mathcal{T})$  and  $s \notin \text{Fn}(\mathcal{T})$  and  $(r_1s_1r_2s_2)^*$  with  $r_1, r_2 \in \text{Fn}(\mathcal{T})$  and  $s_1, s_2 \notin \text{Fn}(\mathcal{T})$ . The following is easy to establish.

**Lemma 3** *Let  $i \in \{1, 2\}$ . It can be decided in PTIME whether a given  $SHOIF_{reg}$ -TBox (resp.  $ALCCOIF_{reg}$ -TBox) is a  $SHOIF_{reg}^i$ -TBox (resp.  $ALCCOIF_{reg}^i$ -TBox).*

We give some examples that illustrate the expressive power of the fragments. All of them fall within  $ALCCOIF_{reg}^1$ .

**Example 4** (1)  $A \sqsubseteq \exists(C? \cdot r)^*.B$  expresses that every  $A$  can reach a  $B$  along an  $r$ -path on which  $C$  is true at every node, akin to the until operator of temporal logic. We can have  $r \in \text{Fn}(\mathcal{T})$  despite the syntactic restrictions.

(2)  $A \sqsubseteq \exists(r+r^-)^*.B$  expresses that whenever  $A$  is true, then  $B$  is true somewhere in the same maximal connected component of the interpretation defined by the role name  $r$ . We can have  $r \in \text{Fn}(\mathcal{T})$ .

(3) Finite satisfiability can be reduced to unrestricted satisfiability in polynomial time in both  $SHOIF_{reg}^1$  and  $SHOIF_{reg}^2$ . Indeed, a  $SHOIF_{reg}$ -TBox  $\mathcal{T}$  is finitely satisfiable iff

$$\mathcal{T}' = \mathcal{T} \cup \{\top \sqsubseteq \exists(r^-)^*.\{a\} \sqcap \exists r^*.\{b\}, \text{func}(r)\}$$

is satisfiable, where the role name  $r$  and individual names  $a, b$  are fresh. This is because in models of  $\mathcal{T}'$ , all domain elements must be on an  $r$ -path from  $a$  to  $b$ , but because  $r$  is functional there can be only one (finite) such path.

(4) A transitive relation can be simulated by the role  $r^+$  and an equivalence relation by the role  $(r+r^-)^*$ . We can enforce that the role name  $r$  is interpreted as a forest (of potentially infinite trees) using the TBox

$$\top \sqsubseteq \exists(r^-)^*.\forall r^-. \perp, \text{func}(r^-).$$

(5) The TBox

$$\top \sqsubseteq \exists r^*.\{a\} \sqcup \exists(r^-)^*.\{a\}, \quad \{a\} \sqsubseteq \neg \exists r^+.\{a\}, \\ \text{func}(r), \quad \text{func}(r^-)$$

enforces that the regular role  $r^+$  is interpreted as a (strict) linear order with successor relation  $r$ .

We next observe that role inclusions can be eliminated by a polynomial time reduction, which relies on the property that functional roles do not have subroles.

**Proposition 5** *Let  $i \in \{1, 2\}$ . (Finite) satisfiability in  $SHOIF_{reg}^i$  can be reduced in polynomial time to (finite) satisfiability in  $ALCCOIF_{reg}^i$ .*

In the remainder of the paper, it will be convenient to work with TBoxes  $\mathcal{T}$  in a certain normal form. A concept definition takes the form  $A \equiv C$  with  $A$  a concept name and  $C$  a potentially compound concept. An interpretation  $\mathcal{I}$  satisfies a concept definition  $A \equiv C$  if  $A^{\mathcal{I}} = C^{\mathcal{I}}$ . Let  $i \in \{1, 2\}$ . An  $ALCCOIF_{reg}^i$ -TBox in normal form is a finite set  $\mathcal{T}$  of concept definitions and functionality assertions such that:

1. every concept definition in  $\mathcal{T}$  is of one of the following forms where  $A, B, B'$  are concept names:

$$A \equiv \{a\} \quad A \equiv \neg B \quad A \equiv B \sqcup B' \\ A \equiv B \sqcap B' \quad A \equiv \exists \alpha.B$$

2. if  $\text{func}(r^-)$  occurs in  $\mathcal{T}$ , then so does  $\text{func}(r)$ , for every role name  $r$ ;

3. if a regular role  $\alpha$  occurs in  $\mathcal{T}$ , then  $\alpha$  satisfies Condition (\* <sub>$i$</sub> ) from Definition 2;

4. for every test  $C?$  that occurs in  $\mathcal{T}$ ,  $C$  is a concept name;

5. if  $\mathcal{T}$  contains  $A \equiv \exists\beta + \beta'.B$ , then  $\mathcal{T}$  contains  $A_1 \equiv \exists\beta.B$ ,  $A_2 \equiv \exists\beta'.B$ , and  $A \equiv A_1 \sqcup A_2$  for some  $A_1, A_2$ ;

6. if  $\mathcal{T}$  contains  $A \equiv \exists\beta \cdot \beta'.B$ , then  $\mathcal{T}$  contains  $A \equiv \exists\beta.A_1$  and  $A_1 \equiv \exists\beta'.B$  for some  $A_1$ ;

7. if  $\mathcal{T}$  contains  $A \equiv \exists\beta^*.B$ , then  $\mathcal{T}$  contains  $A \equiv B \sqcup A_1$  and  $A_1 \equiv \exists\beta.A$  for some  $A_1$ ;

8. if  $\mathcal{T}$  contains  $A \equiv \exists B_1?.B_2$ , then also  $A \equiv B_1 \sqcap B_2 \in \mathcal{T}$ .

Note that Condition 3 above is stronger than what is required by Definition 2 since (\* <sub>$i$</sub> ) needs to be satisfied for every regular role  $\alpha$  that occurs in  $\mathcal{T}$  rather than only for those  $\alpha$  that occur in the form  $\alpha^*$ . Conditions (5)-(8) are inspired by the Fischer-Ladner closure in PDL (Fischer and Ladner 1979). Every  $ALCCOIF_{reg}^i$ -TBox in normal form can be viewed as an  $ALCCOIF_{reg}^i$ -TBox according to the original definition by

reading concept definitions  $A \equiv C$  as an abbreviation for the  $\text{CI} \sqcap \sqsubseteq (\neg A \sqcup C) \sqcap (\neg C \sqcup A)$ . The following lemma shows that we can work with TBoxes in normal form without loss of generality.

**Proposition 6** *Every  $\text{ALCOI}\mathcal{F}_{\text{reg}}^i$ -TBox  $\mathcal{T}$  can be converted in polynomial time into an  $\text{ALCOI}\mathcal{F}_{\text{reg}}^i$ -TBox  $\mathcal{T}'$  in normal form such that  $\mathcal{T}$  is (finitely) satisfiable iff  $\mathcal{T}'$  is.*

## 4 Tree Automata with Parikh Constraints

We define tree automata with existential Presburger constraints on the Parikh image as a central tool for our decision procedures. These are related to Parikh automata on finite words as used in (Klaedtke and Rueß 2003; Figueira and Libkin 2015; Filiot, Guha, and Mazzocchi 2019). Our automata may run on finite trees or on infinite trees.

A tree  $T$  is a prefix-closed (finite or infinite) set of words over the infinite alphabet  $\mathbb{N} \setminus \{0\}$ . A node  $w \in T$  is a successor of  $v \in T$  and  $v$  is a predecessor of  $w$  if  $w = v \cdot i$  for some  $i \in \mathbb{N}$  and where ‘ $\cdot$ ’ denotes concatenation. As a convention, we write  $w \cdot (-1)$  to denote the predecessor of  $w$ . Moreover,  $w$  is a descendant of  $v \in T$  if  $w = v \cdot u$  for some  $u \in \mathbb{N}^*$ . A tree is  $k$ -ary if every node has exactly  $k$  or zero successors. Let  $\Sigma$  be a finite alphabet. A  $\Sigma$ -labeled tree is a pair  $(T, \tau)$  where  $T$  is a tree and  $\tau : T \rightarrow \Sigma$ . We may write only  $\tau$  for the  $\Sigma$ -labeled tree  $(T, \tau)$  if  $T$  is understood.

Presburger arithmetic is the first-order (FO) theory of the non-negative integers with addition (Haase 2018; Presburger 1929). We also consider a version that adds  $\aleph_0$ . A term is a constant 0, 1, or  $\aleph_0$ , a variable, or of the form  $t_1 + t_2$  with  $t_1, t_2$  terms. A Presburger formula is an FO formula over atoms of the form  $t_1 = t_2$  with  $t_1, t_2$  terms. It is existential if it takes the form  $\varphi(\mathbf{x}) = \exists \mathbf{y} \psi(\mathbf{x}, \mathbf{y})$  where  $\psi$  is quantifier-free and  $\mathbf{x}, \mathbf{y}$  denote tuples of variables. We say that  $\mathbf{a} \in (\mathbb{N} \cup \{\aleph_0\})^{|\mathbf{x}|}$  satisfies  $\varphi$  if  $(\mathbb{N} \cup \{\aleph_0\}, 0, 1, \aleph_0, +) \models \varphi(\mathbf{a})$  where  $|\mathbf{x}|$  denotes the length of  $\mathbf{x}$ . Further,  $\varphi$  is satisfiable over  $\mathbb{N} \cup \{\aleph_0\}$  if there exists a satisfying such  $\mathbf{a}$ . For Presburger sentences that do not use the constant  $\aleph_0$ , satisfaction by  $\mathbf{a} \in \mathbb{N}^{|\mathbf{x}|}$  and satisfiability over  $\mathbb{N}$  are defined accordingly.

**Definition 7** A two-way alternating Büchi tree automaton with existential Presburger constraints on the Parikh image ( $2\text{ABTA}_{\exists PP}$ ) on  $k$ -ary trees is a tuple  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$  where  $Q$  is a finite set of states,  $\Sigma = \{\sigma_1, \dots, \sigma_m\}$  a finite input alphabet,  $q_0 \in Q$  the initial state,  $\delta$  a transition function,  $\Omega \subseteq Q$  a set of recurring states, and  $\varphi_c$  an existential Presburger formula with free variables  $x_{\sigma_1}, \dots, x_{\sigma_m}$ . The transition function  $\delta$  maps each state  $q$  and input letter  $\sigma \in \Sigma$  to a positive Boolean formula  $\delta(q, \sigma)$  over the truth constants true and false and transitions of the form  $p, \diamond p, \square p, \diamond^- p, \square^- p$  with  $p \in Q$ .

A run of a  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$  on a  $k$ -ary  $\Sigma$ -labeled tree  $(T, \tau)$  is a  $T \times Q$ -labeled tree  $(T_r, r)$  such that  $r(\varepsilon) = (\varepsilon, q_0)$  and whenever  $x \in T_r$ ,  $r(x) = (n, q)$ , and  $\delta(q, \tau(n)) = \theta$ , then there is a subset  $S$  of the transitions that occur in  $\theta$  such that  $S$  satisfies  $\theta$  and:

- for every  $p \in S$ , there is a successor  $x'$  of  $x$  in  $T_r$  with  $r(x') = (n, p)$ ;

- for every  $\diamond p \in S$ , there is an  $i \in \{1, \dots, k\}$  and a successor  $x'$  of  $x$  in  $T_r$  with  $r(x') = (n \cdot i, p)$ ;
- for every  $\square p \in S$  and all successors  $n \cdot i \in T$  of  $n$ , there is a successor  $x'$  of  $x$  in  $T_r$  with  $r(x') = (n \cdot i, p)$ ;
- for every  $\diamond^- p \in S$ , there is a successor  $x'$  of  $x$  in  $T_r$  with  $r(x') = (n \cdot (-1), p)$ ;
- if  $n \cdot (-1)$  is defined, then for every  $\square^- p \in S$ , there is a successor  $x'$  of  $x$  in  $T_r$  with  $r(x') = (n \cdot (-1), p)$ .

The run  $(T_r, r)$  is accepting if for every infinite path  $\gamma = i_0 i_1 \dots$  in  $T_r$ , there is a  $q \in \Omega$  such that for infinitely many  $j$ ,  $r(i_j)$  takes the form  $(n, q)$  for some  $n$ . We say that  $\mathfrak{A}$  accepts  $(T, \tau)$  if

1. there exists an accepting run of  $\mathfrak{A}$  on  $(T, \tau)$ ;
2.  $\#\_{\sigma_1}(\tau), \dots, \#\_{\sigma_m}(\tau)$  satisfies  $\varphi_c$  over  $\mathbb{N} \cup \{\aleph_0\}$

where  $\#\_{\sigma}(\tau)$  is the cardinality of  $\{n \in T \mid \tau(n) = \sigma\}$ . We use  $L(\mathfrak{A})$  to denote the set of all  $k$ -ary  $\Sigma$ -labeled trees accepted by  $\mathfrak{A}$  and  $L_f(\mathfrak{A})$  for the set of all finite such trees.

Let us briefly discuss the relation to Parikh automata as originating from (Klaedtke and Rueß 2003). Apart from running on finite words rather than on infinite trees, the automata of Klaedtke and Rueß use Presburger constraints in a more general way, maintaining a vector over  $\mathbb{N}$  along with the run and checking at the end whether the obtained vector is a model of a (not necessarily existential) Presburger formula. On the other hand, our automata are more general in that they are two-way and alternating. For the two-way version of the automata defined by Klaedtke and Rueß, emptiness is actually undecidable (Filiot, Guha, and Mazzocchi 2019).

We mainly need two properties of  $2\text{ABTA}_{\exists PP}$ : that their non-emptiness problem is decidable in NEXPTIME and that they are closed under intersection with very modest blowup of the automaton size. The latter is essentially a consequence of the fact that the same is true for standard two-way alternating Büchi tree automata.

**Lemma 8** *For  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A}_i = (Q_i, \Sigma, q_{0,i}, \delta_i, \Omega_i, \varphi_{c,i})$ ,  $i \in \{1, 2\}$ , one can construct in time polynomial in  $\|\mathfrak{A}_1\| + \|\mathfrak{A}_2\|$  a  $2\text{ABTA}$   $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$  such that  $L(\mathfrak{A}) = L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$ ,  $|Q| = |Q_1| + |Q_2| + 1$ , and  $\varphi_c = \varphi_{c,1} \wedge \varphi_{c,2}$ .*

The non-emptiness (resp. finite non-emptiness) problem for  $2\text{ABTA}_{\exists PP}$  is to decide, given a  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A}$ , whether  $L(\mathfrak{A}) \neq \emptyset$  (resp.  $L_f(\mathfrak{A}) \neq \emptyset$ ).

**Theorem 9** *Non-emptiness of  $2\text{ABTA}_{\exists PP}$  is decidable in NEXPTIME. More precisely, there is a non-deterministic algorithm that, given a  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$ , decides whether  $L(\mathfrak{A}) \neq \emptyset$  and runs in time single exponential in  $|Q|$  and polynomial in  $|\Sigma| + \|\delta\| + \|\varphi_c\|$ . The same is true for finite non-emptiness.*

The proof of Theorem 9 is presented in the subsequent sections. It also shows that decidability of non-emptiness extends to the parity acceptance condition and to the case where constraints on Parikh images are unrestricted Presburger formulas rather than existential ones. We leave such generalizations and the precise complexity of non-emptiness for  $2\text{ABTA}_{\exists PP}$  as future work.

## 5 Non-Emptiness of 2ABTA<sub>∃PP</sub>: Finite Case

We first establish the finite version of Theorem 9. Although it can be viewed as a special case of the infinite version, its proof is much simpler than in the infinite case and showcases the central idea in a clearer way.

A *two-way alternating Büchi tree automata (2ABTA)* is a 2ABTA<sub>∃PP</sub> in which the set of Presburger constraints is empty. A central observation is the following.

**Lemma 10** *Let  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$  be a 2ABTA with  $\Sigma = \{\sigma_1, \dots, \sigma_n\}$ . We can construct in time single exponential in  $|Q|$  and polynomial in  $|\Sigma| + \|\delta\|$  an existential Presburger formula  $\varphi_{\mathfrak{A}}$  with free variables  $x_{\sigma_1}, \dots, x_{\sigma_n}$  such that for all  $\mathbf{a} \in \mathbb{N}^n$ , the following are equivalent:*

1.  $\mathbf{a}$  satisfies  $\varphi_{\mathfrak{A}}$  over  $\mathbb{N}$ ;
2.  $L_f(\mathfrak{A})$  contains a  $(T, \tau)$  with  $\mathbf{a} = \#\sigma_1(\tau), \dots, \#\sigma_n(\tau)$ .

Lemma 10 immediately yields the upper bound for finite non-emptiness in Theorem 9. Let a 2ABTA<sub>∃PP</sub>  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$  be given,  $\Sigma = \{\sigma_1, \dots, \sigma_n\}$ , and let  $\mathfrak{A}'$  be the 2ABTA obtained from  $\mathfrak{A}$  by dropping the last component. We construct the formula  $\varphi_{\mathfrak{A}'}$  with free variables  $x_{\sigma_1}, \dots, x_{\sigma_n}$  from Lemma 10. Then the Presburger sentence

$$\exists x_{\sigma_1} \cdots \exists x_{\sigma_n} (\varphi_{\mathfrak{A}'} \wedge \varphi_c)$$

is satisfiable over  $\mathbb{N}$  iff  $L_f(\mathfrak{A}) \neq \emptyset$ , which we can verify in NP (Verma, Seidl, and Schwentick 2005; Haase 2018).

It thus remains to prove Lemma 10. A *non-deterministic top-down automaton on  $k$ -ary finite trees (NTA)* is a tuple  $\mathfrak{A} = (Q, \Sigma, q_0, \Delta)$ , where  $Q$  is a set of *states*,  $q_0 \in Q$  is the *initial state*, and  $\Delta \subseteq (Q \times \Sigma) \cup (Q \times \Sigma \times Q^k)$  is the *transition relation*. A transition  $(q, \sigma, q_1, \dots, q_k)$  means that when  $\mathfrak{B}$  is in state  $q$  and reads symbol  $\sigma$ , then it can send the states  $q_1, \dots, q_k$  to the  $k$  successors of the current node. A transition  $(q, \sigma)$  means that  $\mathfrak{B}$  can stop at a leaf when in state  $q$ . The formal definition of runs is in the appendix.

It is well-known that every 2ABTA  $\mathfrak{A}$  can be translated into an NTA  $\mathfrak{B}$  that accepts the same language, incurring at most a single exponential blow-up in the number of states (Slutzki 1985; Vardi 1998). We can view  $\mathfrak{B}$ , in turn, as a *context free grammar (CFG)  $G$*  by interpreting the states as non-terminal symbols with the initial state being the start symbol, the symbols from the input alphabet  $\Sigma$  as the terminal symbols, and each transition  $(q, \sigma, q_1, \dots, q_m)$  as a grammar rule  $q \rightarrow \sigma q_1 \cdots q_m$ . Then for any tree  $(T, \tau) \in L(\mathfrak{B})$ , there is a word  $w \in L(G)$  such that  $\#\sigma(\tau) = \#_{\sigma}(w)$  for all  $\sigma \in \Sigma$ , and vice versa. Now, the following yields Lemma 10.

**Theorem 11** (Verma, Seidl, and Schwentick 2005, Theorem 4) *Given a CFG  $G$  with terminals  $\Sigma = \{\sigma_1, \dots, \sigma_n\}$ , one can compute in linear time an existential Presburger formula  $\varphi_G(x_{\sigma_1}, \dots, x_{\sigma_n})$  such that for all  $\mathbf{a} \in \mathbb{N}^n$ ,  $\mathbf{a}$  satisfies  $\varphi_G$  over  $\mathbb{N}$  iff there is a word  $w \in L(G)$  such that  $\mathbf{a} = \#\sigma_1(w), \dots, \#\sigma_n(w)$ .<sup>1</sup>*

<sup>1</sup>A small correction to the construction from (Verma, Seidl, and Schwentick 2005) is in the long version of (Hague and Lin 2012).

## 6 Non-Emptiness of 2ABTA<sub>∃PP</sub>: Infinite Case

To adapt the approach used in the previous section to infinite non-emptiness, we would need a counterpart of Lemma 10 that quantifies over all  $\mathbf{a} \in (\mathbb{N} \cup \{\aleph_0\})^n$  and uses  $L(\mathfrak{A})$  in place of  $L_f(\mathfrak{A})$  in Point 2. However, it is not clear how to construct the required Presburger formula  $\varphi_{\mathfrak{A}}$  then; clearly, we can no longer proceed via CFGs as in the finite case. We thus use a different approach, based on the following.

**Lemma 12** *Let  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$  be a 2ABTA and let  $\Sigma_{fin} = \{\sigma_1, \dots, \sigma_n\} \subseteq \Sigma$ . There is a finite set  $\mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  of existential Presburger formulas with free variables  $x_{\sigma_1}, \dots, x_{\sigma_n}$  such that for all  $\mathbf{a} \in \mathbb{N}^n$ , the following are equivalent:*

1.  $\mathbf{a}$  satisfies some  $\varphi \in \mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  over  $\mathbb{N}$ ;
2. there is a  $(T, \tau) \in L(\mathfrak{A})$  such that
  - (a)  $\mathbf{a} = \#\sigma_1(\tau), \dots, \#\sigma_n(\tau)$ ;
  - (b)  $\#\sigma(\tau) = \aleph_0$  for all  $\sigma \in \Sigma \setminus \Sigma_{fin}$ .

Moreover, there is a non-deterministic procedure that given  $\mathfrak{A}$  and  $\Sigma_{fin}$  generates exactly the formulas in  $\mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  as possible outputs, running in time single exponential in  $|Q|$  and polynomial in  $\|\delta\| + |\Sigma|$ .

Before proving Lemma 12, we first observe that it yields the upper bound in Theorem 9. Let a 2ABTA<sub>∃PP</sub>  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$  be given,  $\Sigma = \{\sigma_1, \dots, \sigma_m\}$ . We decide whether  $L(\mathfrak{A}) = \emptyset$  using the following non-deterministic procedure. We first guess a set  $\Sigma_{fin} = \{\sigma_{i_1}, \dots, \sigma_{i_n}\} \subseteq \Sigma$  and use the non-deterministic procedure from Lemma 12 to guess a sentence  $\psi \in \mathcal{P}_{\mathfrak{A}', \Sigma_{fin}}$  with free variables  $x_{\sigma_{i_1}}, \dots, x_{\sigma_{i_n}}$  where  $\mathfrak{A}'$  is the 2ABTA obtained from  $\mathfrak{A}$  by dropping the last component. We then construct the Presburger sentence

$$\exists x_{\sigma_1} \cdots \exists x_{\sigma_m} (\psi \wedge \varphi_c \wedge \bigwedge_{\sigma \in \Sigma_{fin}} x_{\sigma} \neq \aleph_0 \wedge \bigwedge_{\sigma \in \Sigma \setminus \Sigma_{fin}} x_{\sigma} = \aleph_0).$$

and check in NP whether it is satisfiable over  $\mathbb{N} \cup \{\aleph_0\}$  NP (Kuncak, Piskac, and Suter 2010). We accept if this is the case. It can be verified that this procedure indeed decides whether  $L(\mathfrak{A}) = \emptyset$  and runs in single exponential time.

We now prove Lemma 12. A *non-deterministic Büchi automaton on  $k$ -ary infinite trees (NBTA)* is a tuple  $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, \Omega)$  where  $(Q, \Sigma, q_0, \Delta)$  is an NTA and  $\Omega \subseteq Q$  is a set of *recurring states*. Runs are defined in the appendix. Note that like 2ABTAs, we define NBTAs so that they accept trees that might have leaves and might even be finite.

Let  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$  be a 2ABTA and let  $\Sigma_{fin} \subseteq \Sigma$ . We can construct in time exponential in  $|Q|$  and polynomial in  $\|\delta\| + |\Sigma|$  an NBTA  $\mathfrak{A}' = (Q', \Sigma, q'_0, \Delta', \Omega')$  such that  $L(\mathfrak{A}') = L(\mathfrak{A})$  (Vardi 1998). We think of trees  $(T, \tau)$  accepted by  $\mathfrak{A}'$  as being partitioned into several components. The *root component* is the finite initial piece of  $T$  that is minimal with the conditions that (1) it contains all occurrences of symbols from  $\Sigma_{fin}$  and (2) all leaf nodes are labeled with symbols from  $\Sigma_{inf} := \Sigma \setminus \Sigma_{fin}$ . Each leaf node of the root component is the root of another component that takes the form of a potentially infinite tree. The root component can be described by an NTA on finite trees and thus

translated into an existential Presburger formula as in the proof of Lemma 10. This allows us to address Condition 2a of Lemma 12. The non-root components can contain only symbols from  $\Sigma_{inf}$ . To satisfy Condition 2b from Lemma 12, each such symbol must occur infinitely often, but not necessarily in each single component. Apart from this, concrete multiplicities are irrelevant for non-root components. In other words, we only care about the existence of certain configurations of non-root components. We describe such configurations by sets of obligations, detailed in the following.

We denote with  $\mathfrak{A}'_q$  the variant of  $\mathfrak{A}'$  that has  $q \in Q'$  as initial state. An *obligation for  $\mathfrak{A}'$  and  $\Sigma_{inf}$*  is a triple  $(q, \sigma, \Psi) \in Q \times \Sigma_{inf} \times 2^{\Sigma_{inf}}$ . We say that  $(q, \sigma, \Psi)$  is *satisfiable* if there is a  $\Sigma_{inf}$ -labeled tree  $(T, \tau) \in L(\mathfrak{A}'_q)$  with  $\tau(\varepsilon) = \sigma$  and  $\#_{\sigma'}(\tau) = \aleph_0$  for all  $\sigma' \in \Psi$ . Intuitively,  $(T, \tau)$  is a non-root component of a tree accepted by  $\mathfrak{A}'$ ; note that symbols from  $\Sigma_{inf} \setminus \Psi$  may occur with any multiplicity. We observe the following.

**Lemma 13** *Given an NBTA  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ , a set  $\Sigma_{inf} \subseteq \Sigma$ , and an obligation  $(q, \sigma, \Psi)$  for  $\mathfrak{A}$  and  $\Sigma_{inf}$ , it is decidable in NP whether  $(q, \sigma, \Psi)$  is satisfiable.*

The proof of Lemma 13 uses a non-emptiness check for NBTA's. In fact, we show that there is a finite set  $\mathbf{B}$  of NBTA's such that  $(q, \sigma, \Psi)$  is satisfiable iff  $L(\mathfrak{A}) \cap L(\mathfrak{B}) \neq \emptyset$  for some  $\mathfrak{B} \in \mathbf{B}$  and we can non-deterministically construct an automaton from  $\mathbf{B}$  in polynomial time.

An *obligation pattern for  $\mathfrak{A}'$  and  $\Sigma_{inf}$*  is a set

$$S = \{(q_1, \sigma_1, \Psi_1), \dots, (q_k, \sigma_k, \Psi_k)\}$$

such that each  $(q_i, \sigma_i, \Psi_i)$  is an obligation for  $\mathfrak{A}'$  and  $\Sigma_{inf}$  with  $\Psi_1, \dots, \Psi_k$  a partition of  $\Sigma_{inf}$ , some  $\Psi_i$  possibly being the empty set. We say that  $S$  is *satisfiable* if every obligation in it is.

The set of formulas  $\mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  from Lemma 12 contains one formula  $\varphi_S$  for each satisfiable obligation pattern  $S = \{(q_1, \sigma_1, \Psi_1), \dots, (q_k, \sigma_k, \Psi_k)\}$  for  $\mathfrak{A}'$  and  $\Sigma_{inf}$ . To define  $\varphi_S$ , we first make precise the NTA that describes root components. For synchronization purposes, its definition depends on  $S$ . Let  $Z = \{(q, \sigma) \mid (q, \sigma, \Psi) \in S \text{ for some } \Psi\}$ . Recall that  $\mathfrak{A}' = (Q', \Sigma, q'_0, \Delta', \Omega')$  is an NBTA. Define the NTA  $\mathfrak{A}_S = (Q', \Sigma_S, q'_0, \Delta_S)$  where  $\Sigma_S = \Sigma \cup Z$  and

$$\Delta_S = \Delta' \cup \{(q, (q, \sigma)) \mid (q, \sigma) \in Z\}.$$

Note that the transitions in  $\Delta_S \setminus \Delta'$  are ‘leaf transitions’, as described in the previous section. It is not difficult to show that  $L(\mathfrak{A}_S)$  consists of all finite  $\Sigma_S$ -labeled trees  $(T, \tau)$  for which there is a  $\Sigma$ -labeled tree  $(T', \tau') \in L(\mathfrak{A}')$  such that

1.  $T \subseteq T'$  and  $\tau'(n) \in \Sigma_{inf}$  for all  $n \in T' \setminus T$ ,
2.  $\tau'(n) = \tau(n)$  for all inner nodes  $n \in T$ , and
3. there is an accepting run  $r$  of the NBTA  $\mathfrak{A}'$  on  $\tau'$  such that  $\tau(n) = (r(n), \tau'(n))$  for every leaf  $n$  of  $T$ .

Let  $\Sigma_{fin} = \{\sigma_1, \dots, \sigma_n\}$  and for uniformity,  $\Sigma_S = \{\sigma_1, \dots, \sigma_{n+m}\}$ . We can view  $\mathfrak{A}_S$  as a CFG  $G$  and obtain a Presburger formula  $\varphi_G$  from Theorem 11 that has free variables  $x_{\sigma_1}, \dots, x_{\sigma_{n+m}}$ . We construct from  $\varphi_G$  the

desired sentence  $\varphi_S$  with free variables  $x_{\sigma_1}, \dots, x_{\sigma_n}$ . For each  $q \in Q'$  and  $\sigma \in \Sigma_{inf}$ , let  $k_{q, \sigma}$  denote the number of triples in  $S$  that are of the form  $(q, \sigma, \Psi)$ . Then

$$\varphi_S = \exists x_{\sigma_{n+1}} \dots \exists x_{\sigma_{n+m}} (\varphi_G \wedge \bigwedge_{(q, \sigma) \in Z} x_{(q, \sigma)} \geq k_{q, \sigma}).$$

We prove in the appendix that the conditions from Lemma 12 are satisfied. Here, we only sketch the non-deterministic procedure that given  $\mathfrak{A}$  and  $\Sigma_{fin}$ , generates the formulas  $\mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  as possible outputs, running in time single exponential in  $|Q|$  and polynomial in  $|\delta| + |\Sigma|$ ; recall that  $Q$  and  $\delta$  are components of the original 2ABTA  $\mathfrak{A}$ . The procedure first constructs the NBTA  $\mathfrak{A}'$  and then guesses an obligation pattern  $S$  for  $\mathfrak{A}'$  and  $\Sigma_{inf}$ . It next uses Lemma 13 to verify that  $S$  is satisfiable and then generates and outputs the formula  $\varphi_S$  as described above. This procedure runs within the required time mainly since the number of obligations in an obligation pattern is bounded by  $(|Q'| + 1) \cdot |\Sigma|$  and is thus polynomial in  $|\Sigma|$  and single exponential in  $|Q|$ .

## 7 The Weaker Fragment

The aim of the section is to establish the following result.

**Theorem 14** *In  $\mathcal{ALCCOLF}_{reg}^1$ , satisfiability and finite satisfiability are NEXPTIME-complete.*

The lower bounds stated in Theorem 14 already apply to  $\mathcal{ALCCOLF}$  and are implicit in (Tobies 2000). It thus suffices to establish the upper bounds. We first give a suitable characterization based on a decomposition of the input TBox and then develop a decision procedure that crucially uses a non-emptiness test for 2ABTA $_{\exists PP}$ .

Let  $\mathcal{T}$  be an  $\mathcal{ALCCOLF}_{reg}^1$ -TBox in normal form. We use

- $\mathcal{T}_{bool}$  to denote the set of concept definitions in  $\mathcal{T}$  that are not of the form  $A \equiv \exists \alpha.B$ ,
- $\mathcal{T}_{reg}$  to denote the set of concept definitions  $A \equiv \exists \alpha.B$  in  $\mathcal{T}$  such that no  $r \in \text{Fn}(\mathcal{T})$  occurs in  $\alpha$ , and
- $\mathcal{T}_r$  to denote the set of concept definitions  $A \equiv \exists \alpha.B$  in  $\mathcal{T}$  such that  $r \in \text{Fn}(\mathcal{T})$  is the only role name that occurs in  $\alpha$ , plus  $\mathcal{T} \cap \{\text{func}(r), \text{func}(r^-)\}$ .

Then  $\mathcal{T}_{bool} \cup \mathcal{T}_{reg}$  is an  $\mathcal{ALCCOL}_{reg}$ -TBox and each TBox  $\mathcal{T}_{bool} \cup \mathcal{T}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , contains no role name other than  $r$ . Due to Condition 2 of TBoxes in normal form,  $r \in \text{Fn}(\mathcal{T})$  implies  $\text{func}(r) \in \mathcal{T}$  (and, potentially,  $\text{func}(r^-) \in \mathcal{T}$ ).

A *type for  $\mathcal{T}$*  is a set  $t$  of concept names used in  $\mathcal{T}$  such that there is a model  $\mathcal{I}$  of  $\mathcal{T}_{bool}$  and a  $d \in \Delta^{\mathcal{I}}$  such that  $t = \{A \text{ used in } \mathcal{T} \mid d \in A^{\mathcal{I}}\}$ . We then also say that  $t$  is the *type realized by  $d$  in  $\mathcal{I}$* , denoted  $\text{tp}_{\mathcal{I}}(d)$ . Let  $\text{tp}(\mathcal{T})$  denote the set of all types for  $\mathcal{T}$ . Let  $\mathcal{I}$  be a model of  $\mathcal{T}_{bool}$ . Then  $\mathcal{I}$  realizes the set of types  $\{\text{tp}_{\mathcal{I}}(d) \mid d \in \Delta^{\mathcal{I}}\}$  and for each  $t \in \text{tp}(\mathcal{T})$ , we denote with  $\#_t(\mathcal{I})$  the cardinality of the set  $\{d \in \Delta^{\mathcal{I}} \mid \text{tp}_{\mathcal{I}}(d) = t\}$ . We now give the characterization.

**Proposition 15** *An  $\mathcal{ALCCOLF}_{reg}^1$ -TBox  $\mathcal{T}$  in normal form is (finitely) satisfiable iff there is a set  $\Gamma \subseteq \text{tp}(\mathcal{T})$  such that*

1. *there is a model  $\mathcal{I}_{reg}$  of  $\mathcal{T}_{bool} \cup \mathcal{T}_{reg}$  that realizes  $\Gamma$ ;*
2. *there are (finite) models  $\mathcal{I}_r$  of  $\mathcal{T}_{bool} \cup \mathcal{T}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , such that for all  $r, s \in \text{Fn}(\mathcal{T})$ , the following conditions hold:*

- (a)  $\mathcal{I}_r$  realizes  $\Gamma$ ;  
(b)  $\#_t(\mathcal{I}_r) = \#_t(\mathcal{I}_s)$  for all  $t \in \Gamma$ .

Notice that the models  $\mathcal{I}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , from Condition 2 of Proposition 15 are synchronized with each other only via the multiplicities of types and with the model  $\mathcal{I}_{reg}$  from Condition 1 only via the realized types. The following lemma is central for addressing Condition 2.

**Proposition 16** *Given an  $\mathcal{ALCCOLF}_{reg}^1$ -TBox  $\mathcal{T}$  in normal form and a set  $\Gamma \subseteq \text{tp}(\mathcal{T})$ , we can construct in time single exponential in  $\|\mathcal{T}\|$  a  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A}_{\mathcal{T}}$  whose number of states is polynomial in  $\|\mathcal{T}\|$  and such that*

1.  $L(\mathfrak{A}_{\mathcal{T}}) \neq \emptyset$  iff Condition 2 of Proposition 15 is satisfied;
2.  $L_f(\mathfrak{A}_{\mathcal{T}}) \neq \emptyset$  iff the finite version of Condition 2 of Proposition 15 is satisfied.

Before we prove Proposition 16, we show how it can be used to establish the upper bounds in Theorem 14. It suffices to guess a set  $\Gamma \subseteq \text{tp}(\mathcal{T})$  and verify Conditions 1 and 2 from Proposition 15. Condition 1 is equivalent to satisfiability of the  $\mathcal{ALCCOL}_{reg}$ -TBox

$$\widehat{\mathcal{T}} = \mathcal{T}_{bool} \cup \mathcal{T}_{reg} \cup \{ \top \sqsubseteq \bigsqcup_{t \in \Gamma} \bigwedge t \} \cup \{ \top \sqsubseteq \exists \widehat{r}. \bigwedge t \mid t \in \Gamma \},$$

where  $\widehat{r}$  is a fresh role name and  $\bigwedge t$  denotes the conjunction that contains all  $A \in t$  and  $\neg A$  for all concept names  $A \notin t$  that occur in  $\mathcal{T}$ . Satisfiability in  $\mathcal{ALCCOL}_{reg}$  is EXPTIME-complete (Bonatti et al. 2008), but we have to be careful as  $\widehat{\mathcal{T}}$  is of size (single) exponential in  $\|\mathcal{T}\|$ . Fortunately, a slight modification of the algorithm in (Bonatti et al. 2008) achieves that the runtime is single exponential only in the number of concept names and concepts of the form  $\exists r.C$  in the input TBox, and the number of such concepts in  $\widehat{\mathcal{T}}$  is polynomial in  $\|\mathcal{T}\|$ . For Condition 2, it suffices to invoke Proposition 16 and Theorem 9.

The rest of the section is devoted to the proof of Proposition 16. Thus let  $\mathcal{T}$  be an  $\mathcal{ALCCOLF}_{reg}^1$ -TBox in normal form and let  $\Gamma \subseteq \text{tp}(\mathcal{T})$ . Take any (finite or infinite) model  $\mathcal{I}_r$  of  $\mathcal{T}_{bool} \cup \mathcal{T}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ . Then  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$  is the graph of a partial function. A crucial observation is that such a graph takes a very special form. In fact, it has at most one directed simple cycle in every connected component and no undirected cycle that is not a directed cycle. It is thus very ‘close’ to a forest and can in fact be turned into a forest by removing at most a single edge from every maximal connected component. Here, ‘forest’ means that the undirected version of  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$  contains no cycle. An example of a component in a finite such graph can be found in Figure 1. If both  $\text{func}(r) \in \mathcal{T}$  and  $\text{func}(r^-) \in \mathcal{T}$ , the form of  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$  is even more restricted and each component is a (potentially two-side infinite) directed path or a directed cycle. Recall that we have excluded the remaining case that  $\text{func}(r^-) \in \mathcal{T}$  and  $\text{func}(r) \notin \mathcal{T}$ . There is no a priori bound on the degree of the graphs  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$ .

We aim at building a  $2\text{ABTA}_{\exists PP}$  that simultaneously verifies the existence of all models  $\mathcal{I}_r$  from Condition 2 of Proposition 15 using a single non-emptiness check. We thus need a way to encode a collection of models  $\mathcal{I}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , as a tree that can be the input to a  $2\text{ABTA}_{\exists PP}$ . Since there

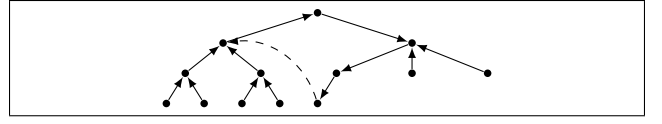


Figure 1: Example component of a graph  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$ ,  $\text{func}(r) \in \mathcal{T}$ . Removing the dashed edge turns it into a tree.

is no bound on the outdegree of these models, we use binary trees and a suitable encoding. Let  $\mathcal{R}$  denote the set  $\{r, r^- \mid r \in \text{Fn}(\mathcal{T})\}$  and  $\mathcal{L} = 2^{\{\text{src}, \text{tgt}\}}$ . To label the trees, we use the alphabet

$$\Sigma = \{\circ\} \cup (\mathcal{R} \times \Gamma \times \mathcal{L}).$$

The symbol ‘ $\circ$ ’ is a label for dummy nodes that we need for the mentioned encoding into binary trees: we simply introduce as many intermediate  $\circ$ -labeled nodes as needed to achieve the required outdegree at each node. For each  $n \in T$  with  $\tau(n) \neq \circ$ , let  $n^\uparrow$  denote the unique  $n' \in T$  (if existing) such that  $n$  is a descendant of  $n'$ ,  $\tau(n') \neq \circ$ , and  $\tau(n'') = \circ$  for all  $n''$  between  $n'$  and  $n$ . If  $\tau(n) = (R, t, L)$  with  $R \in \{r, r^-\}$ , then  $n$  describes a domain element in the interpretation  $\mathcal{I}_r$  that realizes type  $t$ . When  $n^\uparrow$  is defined, then additionally  $(n^\uparrow, n) \in R^{\mathcal{I}_r}$ . The src and tgt markers are used to represent extra edges that close the single cycles in maximal connected components of graphs  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$  as discussed above. A *semi-root* is a node  $n \in T$  such that  $\tau(n) \neq \circ$  and  $n^\uparrow$  is undefined. A  $\Sigma$ -labeled tree  $(T, \tau)$  is *well-formed* if it satisfies the following conditions:

1. for each  $r \in \text{Fn}(\mathcal{T})$ , there is at least one node  $n \in T$  such that  $\tau(n) = (R, t, L)$  with  $R \in \{r, r^-\}$ ;
2. if  $\tau(n) = (R, t, L)$ , then there is no descendant  $n'$  of  $n$  with  $\tau(n') = (R', t', L')$  such that the role names in  $R$  and  $R'$  are different;
3. for each nominal  $\{a\}$  in  $\mathcal{T}$  and  $r \in \text{fn}(\mathcal{T})$ , there is a unique  $n_{a,r} \in T$  such that  $\tau(n_{a,r}) = (R, t, L)$  with  $R \in \{r, r^-\}$  and  $A \in t$  for some  $A \equiv \{a\} \in \mathcal{T}$ ;
4. for every  $r \in \text{Fn}(\mathcal{T})$  and semi-root  $n \in T$ , either there is a unique descendant  $n_1$  of  $n$  with  $\tau(n_1) = (R, t_1, L_1)$  and  $\text{src} \in L_1$  and a unique descendant  $n_2$  of  $n$  with  $\tau(n_2) = (R_2, t_2, L_2)$  and  $\text{tgt} \in L_2$ , or there is no descendant of either kind.

Condition 1 ensures that the interpretations  $\mathcal{I}_r$  have non-empty domains. Condition 2 separates the representations of the interpretations  $\mathcal{I}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ : each maximal connected component of an  $\mathcal{I}_r$  is represented in a subtree rooted at a different semi-root. Condition 4 ensures that there is at most one extra edge per maximal connected component of  $\mathcal{I}_r$ , for each  $r \in \text{Fn}(\mathcal{T})$ .

Every well-formed  $\Sigma$ -labeled tree  $(T, \tau)$  gives rise to a collection of interpretations  $\mathcal{I}_r^T$ ,  $r \in \text{Fn}(\mathcal{T})$ , as follows:

$$\begin{aligned} \Delta^{\mathcal{I}_r^T} &= \{n \in T \mid \tau(n) = (R, t, L), R \in \{r, r^-\}\} \\ a^{\mathcal{I}_r^T} &= n_{a,r} \\ A^{\mathcal{I}_r^T} &= \{n \mid \tau(n) = (R, t, L), R \in \{r, r^-\}, \text{ and } A \in t\} \\ r^{\mathcal{I}_r^T} &= \{(n^\uparrow, n) \mid \tau(n) = (r, t, L)\} \cup \\ &\quad \{(n, n^\uparrow) \mid \tau(n) = (r^-, t, L)\} \cup \end{aligned}$$

$$\begin{aligned} & \{(n_1, n_2) \mid \tau(n_i) = (R_i, t_i, L_i), i \in \{1, 2\}, \\ & \quad R_1, R_2 \in \{r, r^-\}, \text{src} \in L_1, \text{tgt} \in L_2, \\ & \quad n_1, n_2 \text{ descendant of same semi-root } n\} \\ & s^{\mathcal{I}_r} = \emptyset \quad \text{if } s \neq r, \end{aligned}$$

for all  $a \in \mathbb{N}_I$ ,  $A \in \mathbb{N}_C$ , and  $s \in \mathbb{N}_R$ . Note that it is not (yet) guaranteed that  $\mathcal{I}_r$  satisfies the CIs and functionality restrictions in  $\mathcal{T}$ , except the CIs in  $\mathcal{T}_{bool}$ . The automaton will verify that this is indeed the case.

Conversely, let  $\mathcal{I}_r$  be a model of  $\mathcal{T}_{bool} \cup \mathcal{T}_r$ , for each  $r \in \text{Fn}(\mathcal{T})$ , such that Conditions 2a and 2b of Proposition 15 are satisfied. Then there is a binary well-formed  $\Sigma$ -labeled tree  $(T, \tau)$  with  $\mathcal{I}_r = \mathcal{I}_r$  for all  $r \in \text{Fn}(\mathcal{T})$ . We construct such a  $(T, \tau)$  that is not necessarily binary, but can easily be made binary by introducing intermediate  $\circ$ -labeled nodes. Recall that the root of any tree  $T$  is  $\varepsilon$ . We put  $\tau(\varepsilon) = \circ$ . As already mentioned, each graph  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$ ,  $r \in \text{Fn}(\mathcal{T})$ , has at most one directed simple cycle in every connected component and no undirected cycle that is not a directed cycle. Let  $m$  be the sum of the numbers of maximal connected components in all graphs  $(\Delta^{\mathcal{I}_r}, r^{\mathcal{I}_r})$ ,  $r \in \text{Fn}(\mathcal{T})$ . Assume that all these components are linearly ordered. For  $1 \leq \ell \leq m$ , we represent the  $\ell$ -th such component  $(\Delta^{\mathcal{J}}, r^{\mathcal{J}})$  using the subtree of  $(T, \tau)$  rooted at the successor  $\ell$  of the root of  $T$ . If  $(\Delta^{\mathcal{J}}, r^{\mathcal{J}})$  has a simple directed cycle, then choose an edge  $(d_0, e_0) \in r^{\mathcal{J}}$  from this cycle and remove it from  $(\Delta^{\mathcal{J}}, r^{\mathcal{J}})$ . Afterwards,  $(\Delta^{\mathcal{J}}, r^{\mathcal{J}})$  has no more cycles and we can thus find a tree  $T_\ell \subseteq \mathbb{N}^*$  and a bijection  $\pi$  between  $T_\ell$  and  $\Delta^{\mathcal{J}}$  such that for all  $n, n' \in T_\ell$ ,  $n'$  is a successor of  $n$  iff  $(\pi(n), \pi(n')) \in r^{\mathcal{J}} \cup (r^-)^{\mathcal{J}}$ . We include in  $T$  all nodes  $\ell n$ ,  $n \in T_\ell$ , and put  $\tau(\ell n i) = (R, t, L)$  where  $R = r$  if  $(\pi(n), \pi(ni)) \in r^{\mathcal{J}}$  and  $R = r^-$  otherwise,  $t = \text{tp}_{\mathcal{J}}(\pi(ni))$ , and  $L$  contains  $\text{src}$  iff  $\pi(ni) = d_0$  and  $\text{tgt}$  iff  $\pi(ni) = e_0$ . We define  $\tau(\ell)$  in the same way using  $\varepsilon$  in place of  $ni$  and setting  $R = r$  ( $R = r^-$  would also work).

The following establishes Proposition 16.

**Lemma 17** *One can construct in time single exponential in  $\|\mathcal{T}\|$  a  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A}_{\mathcal{T}}$  whose number of states is polynomial in  $\|\mathcal{T}\|$  and such that  $L(\mathfrak{A}_{\mathcal{T}})$  consists of exactly those binary well-formed  $\Sigma$ -labeled trees  $(T, \tau)$  such that for  $\mathcal{I}_r = \mathcal{I}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , Condition 2 of Proposition 15 is satisfied.*

**Proof.** (sketch) The  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A}_{\mathcal{T}}$  is the intersection of several  $2\text{ABTA}_{\exists PP}$ , namely  $\mathfrak{A}_1, \mathfrak{A}_2, \mathfrak{A}_3$ , and  $\mathfrak{A}_r$  for each  $r \in \text{Fn}(\mathcal{T})$ . While  $\mathfrak{A}_1$  checks that  $(T, \tau)$  is well-formed,  $\mathfrak{A}_2$  verifies that the  $\mathcal{I}_r$  satisfy Conditions 2a and 2b from Proposition 15 and  $\mathfrak{A}_3$  makes sure that  $R^{\mathcal{I}_r}$  is a partial function whenever  $\text{func}(R) \in \mathcal{T}$ ,  $R \in \{r, r^-\}$ .  $\mathfrak{A}_1$  and  $\mathfrak{A}_2$  are straightforward to construct, in  $\mathfrak{A}_2$  we use Presburger constraints to ensure Conditions 2a and 2b from Proposition 15.  $\mathfrak{A}_3$  is implemented by making sure that certain forbidden patterns do not occur in  $(T, \tau)$  such as:

- a node  $n \in T$  such that  $\tau(n) = (r, t_n, L_n)$ ,  $\tau(n^\uparrow) = (r^-, t_{n^\uparrow}, L_{n^\uparrow})$ , and  $n^\uparrow$  defined;
- a node  $n \in T$  such that  $\tau(n) = (r, t, L)$ ,  $\text{src} \in L$ , and  $n^\uparrow$  defined;
- nodes  $n_1, n_2$  such that  $\tau(n_i) = (r^-, t_i, L_i)$  for  $i \in \{1, 2\}$ ,  $\text{func}(r^-) \in T$ , and  $n_1^\uparrow = n_2^\uparrow$ .

Working out a complete list of patterns is straightforward.

The purpose of each automaton  $\mathfrak{A}_r$  is to ensure that  $\mathcal{I}_r$  satisfies the CIs from  $\mathcal{T}_r$ . It sends a copy of itself to every node  $n$  of the input tree  $(T, \tau)$  and verifies that when  $\tau(n) = (R, t, L)$  with  $R \in \{r, r^-\}$  and  $A \equiv \alpha.B \in \mathcal{T}_r$ , then  $A \in t$  iff  $n \in (\exists \alpha.B)^{\mathcal{I}_r}$ . The automaton thus needs to verify the (non-)existence of an  $\alpha$ -path that starts at  $n$  and ends in a node whose type includes  $B$ . This is implemented by representing  $\alpha$  as a finite automaton  $\mathcal{B}$  on finite words, tracing runs of  $\mathcal{B}$  through the  $\mathcal{I}_r$  part of  $(T, \tau)$ , and making use of the Büchi acceptance condition.  $\square$

This finishes the proof of the upper bound in Theorem 14. It is a noteworthy consequence of our proof and bounds on the sizes of solutions of existential Presburger formulas (Papadimitriou 1981) that every finitely satisfiable  $\text{SHOLIF}_{reg}^1$ -TBox  $\mathcal{T}$  has a model of size at most double exponential in  $\|\mathcal{T}\|$ . This bound is tight since there are  $\text{ALCOLF}$ -TBoxes that enforce models of double exponential size.

## 8 The Stronger Fragment

The aim of the section is to establish the following result.

**Theorem 18** *In  $\text{ALCOLF}_{reg}^2$ , satisfiability and finite satisfiability are decidable in  $2\text{NEXPTIME}$ .*

A  $\text{NEXPTIME}$  lower bound is obtained from Theorem 14, but the precise complexity remains open. In the upper bound proof, it is no longer possible to separate the input TBox  $\mathcal{T}$  into components  $\mathcal{T}_{bool}, \mathcal{T}_{reg}, \mathcal{T}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , since regular expressions that contain a role name  $r \in \text{Fn}(\mathcal{T})$  might now contain also other role names. It is thus not clear how to give a characterization in the style of Proposition 15 and we resort to directly encoding (finite) satisfiability as a non-emptiness check of a suitably constructed  $2\text{ABTA}_{\exists PP}$ .

**Proposition 19** *Given an  $\text{ALCOLF}_{reg}^2$ -TBox  $\mathcal{T}$  in normal form, we can construct in time single exponential in  $\|\mathcal{T}\|$  a  $2\text{ABTA}_{\exists PP}$   $\mathfrak{A}_{\mathcal{T}}$  such that*

1.  $L(\mathfrak{A}_{\mathcal{T}}) \neq \emptyset$  iff  $\mathcal{T}$  is satisfiable;
2.  $L_f(\mathfrak{A}_{\mathcal{T}}) \neq \emptyset$  iff  $\mathcal{T}$  is finitely satisfiable.

To establish Proposition 19, let  $\mathcal{T}$  be an  $\text{ALCOLF}_{reg}^2$ -TBox in normal form whose satisfiability is to be decided. We use  $\Sigma$ -labeled trees for the alphabet  $\Sigma = \{\circ\} \cup (\mathcal{R} \times \text{tp}(\mathcal{T}) \times \mathcal{L})$ ,  $\mathcal{R}$  and  $\mathcal{L}$  defined as in the previous section and where types now have to satisfy an additional *progress condition* spelled out in detail in the appendix. We also use the same notion of well-formedness as in the previous section, but extended with the following condition, similar to Condition 2b of Proposition 15:

5. for all  $t \in \text{tp}(\mathcal{T})$  and  $r_1, r_2 \in \text{Fn}(\mathcal{T})$ ,

$$\sum_{L \in \mathcal{L}, R \in \{r_1, r_1^-\}} \#_{(R, t, L)}(\tau) = \sum_{L \in \mathcal{L}, R \in \{r_2, r_2^-\}} \#_{(R, t, L)}(\tau).$$

In the previous section, a  $\Sigma$ -labeled tree  $(T, \tau)$  encodes a collection of interpretations  $\mathcal{I}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ . Here, we want  $(T, \tau)$  to encode a single interpretation  $\mathcal{I}_\tau$  that is a model of  $\mathcal{T}$ . Essentially, what were distinct interpretations  $\mathcal{I}_r$  previously are now ‘slices’ of  $\mathcal{I}_\tau$ . In particular, there is no



one-to-one correspondence between the nodes in  $(T, \tau)$  and the domain elements in  $\mathcal{I}_\tau$  as every domain element of  $\mathcal{I}_\tau$  is represented by multiple nodes in  $(T, \tau)$ , one for each  $r \in \text{Fn}(\mathcal{T})$  (each ‘slice’). We next make this precise.

A well-formed  $\Sigma$ -labeled tree  $(T, \tau)$  gives rise to an interpretation  $\mathcal{I}_\tau$  as follows. For each  $r \in \text{Fn}(\mathcal{T})$ , let  $T|_r$  denote the set of nodes  $n \in T$  such that  $\tau(n) = (R, t, L)$  with  $R \in \{r, r^-\}$ . Choose an  $r_0 \in \text{Fn}(\mathcal{T})$  (which we can w.l.o.g. assume to exist). By Condition 5 above, we can choose a bijection  $\pi_r$  from  $T|_{r_0}$  to  $T|_r$  for every  $r \in \text{Fn}(\mathcal{T})$  such that for all  $n \in T|_{r_0}$  and  $r \in \text{Fn}(\mathcal{T})$ ,  $\tau(n) = (R, t, L)$  implies that  $\tau(\pi_r(n))$  is of the form  $(R', t, L')$ . Now define

$$\begin{aligned} \Delta^{\mathcal{I}_\tau} &= T|_{r_0} \\ a^{\mathcal{I}_\tau} &= n_{a, r_0} \\ A^{\mathcal{I}_\tau} &= \{n \mid \tau(n) = (R, t, L) \text{ and } A \in t\} \\ r^{\mathcal{I}_\tau} &= \{(n^\uparrow, n) \mid \tau(\pi_r(n)) = (r, t, L)\} \cup \\ &\quad \{(n, n^\uparrow) \mid \tau(\pi_r(n)) = (r^-, t, L)\} \cup \\ &\quad \{(n_1, n_2) \mid \tau(\pi_r(n_i)) = (R_i, t_i, L_i), i \in \{1, 2\}, \\ &\quad \quad R_1, R_2 \in \{r, r^-\}, \text{src} \in L_1, \text{tgt} \in L_2, \\ &\quad \quad n_1, n_2 \text{ descendant of same semi-root } n\} \end{aligned}$$

for all  $a \in \mathbb{N}_I$ ,  $A \in \mathbb{N}_C$ , and  $r \in \text{Fn}(\mathcal{T})$ . We still have to define  $s^{\mathcal{I}_\tau}$  for all  $s \in \mathbb{N}_R \setminus \text{Fn}(\mathcal{T})$ . A crucial idea, also used in the proof of Proposition 15, is to interpret these role names in a maximal way. More precisely, we define  $s^{\mathcal{I}_\tau}$  to contain all pairs  $(n_1, n_2)$ ,  $\tau(n_i) = (R_i, t_i, L_i)$  for  $i \in \{1, 2\}$ , such that the following conditions are satisfied:

1. for all CIs  $A \equiv \exists s. B \in \mathcal{T}$ ,  $B \in t_2$  implies  $A \in t_1$ ;
2. for all CIs  $A \equiv \exists s^-. B \in \mathcal{T}$ ,  $B \in t_1$  implies  $A \in t_2$ .

Conversely, we can encode every model  $\mathcal{I}$  of  $\mathcal{T}$  as a well-formed binary  $\Sigma$ -labeled tree  $(T, \tau)$  with  $\mathcal{I}_\tau = \mathcal{I}$ . This is done as in the previous section, using  $\mathcal{I}$  in place of each interpretation  $\mathcal{I}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ —the extension  $r'^{\mathcal{I}}$  of all role names  $r' \neq r$  is simply ignored when using  $\mathcal{I}$  as  $\mathcal{I}_r$ . Role names  $s \in \mathbb{N}_R \setminus \text{Fn}(\mathcal{T})$  are not represented explicitly in  $(T, \tau)$ , but assumed to be interpreted in a maximal way as detailed above. The following yields Proposition 19.

**Lemma 20** *One can construct in time single exponential in  $\|\mathcal{T}\|$  a 2ABTA $_{\exists PP}$   $\mathfrak{A}_\mathcal{T}$  such that  $L(\mathfrak{A}_\mathcal{T})$  consists of exactly those binary well-formed  $\Sigma$ -labeled trees  $(T, \tau)$  with  $\mathcal{I}_\tau \models \mathcal{T}$ .*

**Proof.** (sketch) We use as  $\mathfrak{A}_\mathcal{T}$  the intersection of 2ABTA $_{\exists PP}$   $\mathfrak{A}_1$ ,  $\mathfrak{A}_2$ , and  $\mathfrak{A}$ .  $\mathfrak{A}_1$  checks that  $\mathcal{I}_\tau$  is well-formed, including the new Condition 5 for which it uses the Presburger constraints.  $\mathfrak{A}_2$  verifies that  $R^{\mathcal{I}_\tau}$  is a partial function whenever  $\text{func}(R) \in \mathcal{T}$ . The construction is similar to those of the automata  $\mathfrak{A}_1$  and  $\mathfrak{A}_3$  in the proof of Lemma 17. The automata  $\mathfrak{A}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , from that proof can no longer be kept separate from each other because we now need to verify the satisfaction of eventualities  $A \equiv \exists \alpha. B$  where  $\alpha$  may contain more than one role  $r$  with  $r \in \text{Fn}(\mathcal{T})$ . We thus replace them by the single automaton  $\mathfrak{A}$ .

The construction of  $\mathfrak{A}$  is similar to that of the automata  $\mathfrak{A}_r$  in the proof of Lemma 17, with some crucial differences.

When verifying the satisfaction of an eventuality  $A \equiv \exists \alpha. B$ , we now might have to traverse multiple roles, functional and non-functional. A central idea is to make the automaton ‘jump’ whenever traversing a role  $R \in \{s, s^-\}$  with  $s \notin \text{Fn}(\mathcal{T})$ . That is, the automaton will non-deterministically move to a (potentially far away) node in the tree whose type  $t_2$  matches with the type  $t_1$  of the current node according to the traversed  $R$  in the sense of Conditions 1 and 2 above. Interpreting role names  $s \notin \text{Fn}(\mathcal{T})$  in a maximal way as explained above ensures that the ‘jump’ corresponds to an actual edge in  $\mathcal{I}_\tau$ . This approach works only because of Condition  $(*_2)$  from the definition of the fragment  $\mathcal{ALCOIF}_{reg}^2$  which makes sure that when following an eventuality  $A \equiv \exists \alpha. B$ , we never have to consecutively traverse  $R_1 \in \{r_1, r_1^-\}$  and  $R_2 \in \{r_2, r_2^-\}$  with  $r_1, r_2 \in \text{Fn}(\mathcal{T})$  distinct without traversing an  $S \in \{s, s^-\}$  with  $s \notin \text{Fn}(\mathcal{T})$  in between. The sandwiched  $S$  allows us to jump and thus to move from the slice of the  $\Sigma$ -labeled tree that represents  $\mathcal{I}_{r_1}$  to the slice that represents  $\mathcal{I}_{r_2}$ , which may be far away. But jumping requires us to memorize types, which leads to exponentially many states.

There is an additional small technical complication related to the ‘slice’ representation of  $\mathcal{I}_\tau$  that manifests at the very beginning of verifying eventualities  $A \equiv \exists \alpha. B$ . We address it using the additional progress condition on types mentioned at the beginning of the section.  $\square$

This finishes the proof of Theorem 18. It follows from our proof that every finitely satisfiable  $\mathcal{SHOIF}_{reg}^2$  TBox  $\mathcal{T}$  has a model of size at most triple exponential in  $\|\mathcal{T}\|$ .

## 9 Conclusions

An important question is whether (finite) satisfiability in unrestricted  $\mathcal{SHOIF}_{reg}$  and  $\mathcal{ALCOIF}_{reg}$  is decidable. However, it appears to be difficult to adapt the presented approach to even modest extensions of  $\mathcal{ALCOIF}_{reg}^1$  and  $\mathcal{ALCOIF}_{reg}^2$ . For example, we do not know how to accommodate local (unqualified) functionality restrictions, that is, concept expressions of the form  $(\leq 1 r)$ . For qualified functionality restrictions  $(\leq 1 r C)$ , it can be observed that this is indeed difficult. Let  $\mathcal{ALCOIF}_{reg, qual}^1$  denote the extension of  $\mathcal{ALCOIF}_{reg}^1$  with restrictions of the latter kind. It turns out that proving decidability of  $\mathcal{ALCOIF}_{reg, qual}^1$  is no easier than proving decidability of unrestricted  $\mathcal{ALCOIF}_{reg}$ .

**Theorem 21** *(Finite) satisfiability in  $\mathcal{ALCOIF}_{reg}$  can be reduced in polynomial time to (finite) satisfiability in  $\mathcal{ALCOIF}_{reg, qual}^1$ .*

It is also interesting to note that adding guarded Boolean operators on roles, as typically indicated by the letter  $b$  in DL names, results in undecidability even when restricted to role names and their inverses. In fact, this extension makes it possible to adapt the undecidability proof for the two-variable guarded fragment of FO with three linear orders that can only be used in guard atoms given in (Kieronski 2011).

## Acknowledgements

Jung and Lutz were supported by ERC consolidator grant 647289 CODA.

## References

- Baader, F.; Horrocks, I.; Lutz, C.; and Sattler, U. 2017. *An Introduction to Description Logic*. Cambridge University Press.
- Baader, F. 1991. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. In *Proceedings of IJCAI*, 446–451. Morgan Kaufmann.
- Bonatti, P. A., and Peron, A. 2004. On the undecidability of logics with converse, nominals, recursion and counting. *Artif. Intell.* 158(1):75–96.
- Bonatti, P. A.; Lutz, C.; Murano, A.; and Vardi, M. Y. 2008. The complexity of enriched  $\mu$ -calculi. *Logical Methods in Computer Science* 4(3).
- Calvanese, D.; Eiter, T.; and Ortiz, M. 2009. Regular path queries in expressive description logics with nominals. In *Proceedings of IJCAI*, 714–720.
- Figueira, D., and Libkin, L. 2015. Path logics for querying graphs: Combining expressiveness and efficiency. In *Proceedings of LICS*, 329–340. IEEE Computer Society.
- Filiot, E.; Guha, S.; and Mazzocchi, N. 2019. Two-way parikh automata. In *Proceedings of FSTTCS*, 40:1–40:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.
- Fischer, M. J., and Ladner, R. E. 1979. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.* 18(2):194–211.
- Fürer, M. 1980. The complexity of the inequivalence problem for regular expressions with intersection. In *Proceedings of ICALP*, 234–245.
- Haase, C. 2018. A survival guide to presburger arithmetic. *SIGLOG News* 5(3):67–82.
- Hague, M., and Lin, A. W. 2012. Synchronisation- and reversal-bounded analysis of multithreaded programs with counters. In *Proceedings of CAV*, volume 7358 of *LNCS*, 260–276. Springer.
- Horrocks, I.; Kutz, O.; and Sattler, U. 2006. The even more irresistible SROIQ. In *Proceedings of KR*, 57–67.
- Keisler, H. J. 1971. *Model Theory for Infinitary Logic: Logic with Countable Conjunctions and Finite Quantifiers*. North Holland Publishing Company.
- Kieronski, E. 2011. Decidability issues for two-variable logics with several linear orders. In *Proceedings of CSL*, 337–351.
- Klaedtke, F., and Rueß, H. 2003. Monadic second-order logics with cardinalities. In *Proceedings of ICALP*, volume 2719 of *LNCS*, 681–696. Springer.
- Kotek, T.; Simkus, M.; Veith, H.; and Zuleger, F. 2015. Extending ALCQIO with trees. In *Proceedings of LICS*, 511–522.
- Kuncak, V.; Piskac, R.; and Suter, P. 2010. Ordered sets in the calculus of data structures. In *Proceedings of CSL*, volume 6247 of *LNCS*, 34–48. Springer.
- Kupferman, O.; Sattler, U.; and Vardi, M. Y. 2002. The complexity of the graded  $\mu$ -calculus. In Voronkov, A., ed., *Proceedings of CADE*, 423–437. Springer.
- Le Duc, C., and Lamolle, M. 2010. Decidability of description logics with transitive closure of roles in concept and role inclusion axioms. In *Proceedings of DL*. CEUR-WS.org.
- Le Duc, C.; Lamolle, M.; and Curé, O. 2013. A decision procedure for *SHOIQ* with transitive closure of roles. In *Proceedings of ISWC*, 264–279. Springer.
- Le Duc, C. 2019. Personal Communication.
- OWL Working Group, W. 2009. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation. Available at <http://www.w3.org/TR/owl2-overview/>.
- Papadimitriou, C. H. 1981. On the complexity of integer programming. *J. ACM* 28(4):765–768.
- Pratt-Hartmann, I. 2007. Complexity of the guarded two-variable fragment with counting quantifiers. *J. Log. Comput.* 17(1):133–155.
- Pratt-Hartmann, I. 2015. The two-variable fragment with counting and equivalence. *Mathematical Logic Quarterly* 61(6):474–515.
- Presburger, M. 1929. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. 92—101.
- Rudolph, S. 2016. Undecidability results for database-inspired reasoning problems in very expressive description logics. In *Proceedings of KR*, 247–257. AAAI Press.
- Sattler, U., and Vardi, M. Y. 2001. The hybrid  $\mu$ -calculus. In *Proceedings of IJCAR*, 76–91. Springer.
- Schwentick, T., and Zeume, T. 2012. Two-variable logic with two order relations. *Logical Methods in Computer Science* 8(1).
- Slutzki, G. 1985. Alternating tree automata. *Theor. Comput. Sci.* 41:305–318.
- Tobies, S. 2000. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *J. Artif. Intell. Res.* 12:199–217.
- Vardi, M. Y. 1998. Reasoning about the past with two-way automata. In *Proceedings of ICALP*, 628–641.
- Verma, K. N.; Seidl, H.; and Schwentick, T. 2005. On the complexity of equational horn clauses. In *Proceedings of CADE*, 337–352. Springer.
- Zeume, T., and Harwath, F. 2016. Order-invariance of two-variable logic is decidable. In *Proceedings of LICS*, 807–816.

## Appendix

### Regular Roles via Finite Automata

For the sake of completeness, we provide the semantics of regular roles. Given an interpretation  $\mathcal{I}$ , every regular role  $\alpha$  induces a binary relation  $\alpha^{\mathcal{I}}$ , defined in the expected way:

$$\begin{aligned} (r^-)^{\mathcal{I}} &= \{(e, d) \mid (d, e) \in r^{\mathcal{I}}\} \\ (C?)^{\mathcal{I}} &= \{(d, d) \mid d \in C^{\mathcal{I}}\} \\ (\alpha^*)^{\mathcal{I}} &= (\alpha^{\mathcal{I}})^* \\ (\alpha_1 \cdot \alpha_2)^{\mathcal{I}} &= \alpha_1^{\mathcal{I}} \circ \alpha_2^{\mathcal{I}} \\ (\alpha_1 + \alpha_2)^{\mathcal{I}} &= \alpha_1^{\mathcal{I}} \cup \alpha_2^{\mathcal{I}} \end{aligned}$$

The interpretation of concepts  $\exists\alpha.C$  is then given by

$$(\exists\alpha.C)^{\mathcal{I}} = \{d \mid \exists e \in C^{\mathcal{I}} \text{ with } (d, e) \in \alpha^{\mathcal{I}}\}.$$

Throughout the appendix, we will often work with a representation of regular roles by finite automata. This is without loss of generality since for every regular expression, we can find in polynomial time an equivalent finite automaton (Fürier 1980). Formally, a *finite automaton* is a tuple  $\mathfrak{B} = (S, \Sigma, s_0, \Delta, F)$ , where  $S$  is a finite set of states,  $s_0 \in S$  is the initial state,  $F \subseteq S$  is a set of accepting states,  $\Delta \subseteq S \times \Sigma \times S$  is the transition relation, and  $\Sigma$  is a finite alphabet. We denote with  $L(\mathfrak{B})$  the set of all words accepted by  $\mathfrak{B}$ , defined as usual.

In the context of regular roles  $\alpha$ , we will consider alphabets  $\Sigma$  with

$$\Sigma \subseteq \{r, r^- \mid r \in \mathbb{N}_R\} \cup \{C? \mid C \text{ an } \mathcal{ALCOIF}_{reg}\text{-concept}\}.$$

Thus, a regular role  $\alpha$  is just a regular expression over some such alphabet. For any interpretation  $\mathcal{I}$ , a finite automaton  $\mathfrak{B}$  over the alphabet  $\Sigma$  induces a binary relation  $E_{\mathfrak{B}}^{\mathcal{I}}$  on  $\Delta^{\mathcal{I}}$  defined by  $(d, e) \in E_{\mathfrak{B}}^{\mathcal{I}}$  iff there is a word  $w = a_1 \dots a_n \in L(\mathfrak{B})$  and a sequence  $d_0, \dots, d_n \in \Delta^{\mathcal{I}}$  with  $d_0 = d, d_n = e$ , and such that for every  $i$  with  $0 \leq i \leq n$ :

- if  $a_i = C?$ , then  $d_{i-1} \in C^{\mathcal{I}}$  and  $d_i = d_{i-1}$ ,
- if  $a_i = r$ , then  $(d_{i-1}, d_i) \in r^{\mathcal{I}}$ , and
- if  $a_i = r^-$ , then  $(d_{i-1}, d_i) \in (r^-)^{\mathcal{I}}$ .

**Lemma 22** *For every regular role  $\alpha$ , we can compute in polynomial time a finite automaton  $\mathfrak{B}$  such that  $\alpha^{\mathcal{I}} = E_{\mathfrak{B}}^{\mathcal{I}}$ , for every interpretation  $\mathcal{I}$ .*

### Proof of Lemma 3

**Lemma 3** *Let  $i \in \{1, 2\}$ . It can be decided in PTIME whether a given  $\mathcal{SHOIF}_{reg}^i$ -TBox (resp.  $\mathcal{ALCOIF}_{reg}$ -TBox) is a  $\mathcal{SHOIF}_{reg}^i$ -TBox (resp.  $\mathcal{ALCOIF}_{reg}^i$ -TBox).*

**Proof.** Condition  $(*_1)$  is a mere syntactic check. For Condition  $(*_2)$ , let  $\mathfrak{B}$  be the finite automaton equivalent to  $\alpha^*$  which exists due to Lemma 22. The condition can now be verified by checking whether there are transitions

$(q, R, q_1), (q_1, C_1?, q_2), \dots, (q_{n-1}, C_{n-1}?, q_n), (q_n, S, q')$  in  $\mathfrak{B}$  with  $R \in \{r, r^-\}$ ,  $S \in \{s, s^-\}$ ,  $r, s \in \text{Fn}(\mathcal{T})$ , and  $r \neq s$ . This boils down to a polynomial number of reachability tests on directed graphs and is thus possible in polynomial time.  $\square$

## Proof of Proposition 5

**Proposition 5** *Let  $i \in \{1, 2\}$ . (Finite) satisfiability in  $\mathcal{SHOIF}_{reg}^i$  can be reduced in polynomial time to (finite) satisfiability in  $\mathcal{ALCOIF}_{reg}^i$ .*

**Proof.** Let  $\mathcal{T}$  be a  $\mathcal{SHOIF}_{reg}^i$ -TBox,  $i \in \{1, 2\}$ . By introducing fresh concept names, we can obtain in polynomial time an equisatisfiable  $\mathcal{SHOIF}_{reg}^i$ -TBox  $\mathcal{T}'$  such that every concept inclusion takes one of the following forms:

$$\begin{aligned} A \sqsubseteq \{a\} \quad \{a\} \sqsubseteq A \quad A \sqsubseteq \neg B \quad \neg A \sqsubseteq B \\ A \sqcap A' \sqsubseteq B \quad A \sqsubseteq \exists\alpha.B \quad \exists\alpha.B \sqsubseteq A \end{aligned}$$

where  $A, A', B$  are concept names and, additionally, all tests  $C?$  are of the form  $A?$  with  $A$  a concept name.

Let  $r$  be a role name. We denote with  $\text{pre}(r)$  the set of all roles  $S$  such that  $\mathcal{T} \models S \sqsubseteq r$ . Now obtain a TBox  $\mathcal{T}''$  from  $\mathcal{T}'$  by:

- dropping all role inclusions,
- replacing in all concept inclusions of the form  $\exists\alpha.B \sqsubseteq A$  every role  $r$  that occurs in  $\alpha$  with  $S_1 + \dots + S_k$  where  $\text{pre}(r) = \{S_1, \dots, S_k\}$ .

Clearly,  $\mathcal{T}''$  can be computed in polynomial time from  $\mathcal{T}'$ .

*Claim.*  $\mathcal{T}''$  is (finitely) satisfiable iff  $\mathcal{T}'$  is (finitely) satisfiable.

*Proof of the Claim.* For “if”, observe that every model of  $\mathcal{T}'$  is also a model of  $\mathcal{T}''$ . For “only if”, let  $\mathcal{I}$  be a model of  $\mathcal{T}''$ . Obtain a new model  $\mathcal{J}$  by setting:

$$\begin{aligned} \Delta^{\mathcal{J}} &= \Delta^{\mathcal{I}} \\ A^{\mathcal{J}} &= A^{\mathcal{I}} \\ r^{\mathcal{J}} &= \bigcup_{S \in \text{pre}(r)} S^{\mathcal{I}} \end{aligned}$$

for all  $A \in \mathbb{N}_C$  and  $r \in \mathbb{N}_R$ . It is routine to verify that  $\mathcal{J}$  is a model of  $\mathcal{T}'$ . In particular, the definition of  $r^{\mathcal{J}}$  does not violate any functionality assertions by our assumption that functional roles do not have subroles. This finishes the proof of the claim.

Observe that  $\mathcal{T}''$  is not necessarily an  $\mathcal{ALCOIF}_{reg}^i$ -TBox. However, the violations occur only in regular expressions on the left-hand side of concept inclusions, which can be removed. Let  $\exists\alpha.B \sqsubseteq A \in \mathcal{T}''$ . Let  $\mathfrak{B} = (S, \Sigma, s_0, \Delta, F)$  be the finite automaton equivalent to  $\alpha$  which exists due to Lemma 22 and let  $S = \{s_0, \dots, s_n\}$ . We introduce fresh concept names  $S_0, \dots, S_n$  and replace the concept inclusion  $\exists\alpha.B \sqsubseteq A \in \mathcal{T}''$  with the following concept inclusions:

$$\begin{aligned} B \sqsubseteq S_0 \\ S_i \sqcap A \sqsubseteq S_j & \quad \text{for all } (s_i, A?, s_j) \in \Delta \\ \exists r^-.S_i \sqsubseteq S_j & \quad \text{for all } (s_i, r, s_j) \in \Delta \\ \exists r.S_i \sqsubseteq S_j & \quad \text{for all } (s_i, r^-, s_j) \in \Delta \\ S_i \sqsubseteq A & \quad \text{for all } s_i \in F \end{aligned}$$

Let  $\mathcal{T}^*$  be the TBox obtained from  $\mathcal{T}''$  by replacing all concept inclusions of the shape  $\exists\alpha.B \sqsubseteq A$  as described above.

Clearly,  $\mathcal{T}^*$  is an  $\mathcal{ALCOIF}_{reg}^i$ -TBox. It is also routine to verify correctness.

*Claim.*  $\mathcal{T}^*$  is (finitely) satisfiable iff  $\mathcal{T}''$  is (finitely) satisfiable.  $\square$

### Proof of Lemma 6

**Proposition 6** *Every  $\mathcal{ALCOIF}_{reg}^i$ -TBox  $\mathcal{T}$  can be converted in polynomial time into an  $\mathcal{ALCOIF}_{reg}^i$ -TBox  $\mathcal{T}'$  in normal form such that  $\mathcal{T}$  is (finitely) satisfiable iff  $\mathcal{T}'$  is.*

**Proof.** Let  $\mathcal{T}$  be an  $\mathcal{ALCOIF}_{reg}^i$ -TBox,  $i \in \{1, 2\}$ . It is easy to achieve that  $\text{func}(r^-) \in \mathcal{T}$  implies  $\text{func}(r) \in \mathcal{T}$ . In fact, if  $\text{func}(r^-) \in \mathcal{T}$  and  $\text{func}(r) \notin \mathcal{T}$ , then we can swap  $r$  and  $r^-$  throughout  $\mathcal{T}$ , which clearly preserves (un)satisfiability.

We proceed in three steps. In the first step, we exhaustively apply the following rewrite steps:

1. replace every concept  $\exists\beta \cdot \beta'.C$  in  $\mathcal{T}$  with  $\exists\beta.\exists\beta'.C$  and
2. replace every concept  $\exists\beta + \beta'.C$  in  $\mathcal{T}$  with  $\exists\beta.A \sqcup \exists\beta'.A$  and add  $A \equiv C$ , with  $A$  a fresh concept name.

The concept name  $A$  in Point 2 above is introduced to avoid an exponential blowup. Call the resulting TBox  $\mathcal{T}_1$ .

In the second step, we introduce a fresh concept name  $X_C$  for every concept  $C$  in  $\mathcal{T}_1$  that is not a concept name. If  $D$  is a concept name, we use  $X_D$  to denote  $D$ . Put

$$\begin{aligned} \delta(A) &= A & \delta(\{a\}) &= X_{\{a\}} \\ \delta(\neg C) &= \neg X_C & \delta(C \sqcap D) &= X_C \sqcap X_D \\ \delta(\exists\alpha.C) &= \exists\beta.X_C \end{aligned}$$

where  $\beta$  is the result of replacing every test  $D?$  in  $\alpha$  that is not nested within another test with  $X_D?$ . Now  $\mathcal{T}_2$  consists of the following:

- $X_{\{a\}} \equiv \{a\}$ , for all nominals  $\{a\}$  that occur in  $\mathcal{T}$ ;
- $X_C \equiv X_D \sqcap A$  for every  $C \sqsubseteq D \in \mathcal{T}_1$ , with  $A$  a fresh concept name;
- $X_C \equiv \delta(C)$  for every concept  $C$  that occurs in  $\mathcal{T}_1$ , possibly as a subconcept.

It can be verified that, at this Point, Conditions 1 to 4 of the normal form are satisfied. For Condition 3, assume that a regular role  $\alpha$  occurs in  $\mathcal{T}_2$ . Then it occurs in a concept  $\exists\alpha.A$ . Due to the first step,  $\alpha$  takes the form  $r$ ,  $r^-$ , or  $\beta^*$ . In all of these cases,  $\alpha$  satisfies Condition  $(*_i)$  from Definition 2 since in  $\mathcal{T}$ , every regular role  $\alpha$  that occurs in the form  $\alpha^*$  does.

The third step ensures Conditions 5 to 7 of TBoxes in normal form by exhaustively applying the following transformations:

1. if  $A \equiv \exists(\beta + \beta').B$  is a concept definition in  $\mathcal{T}_2$ , then add  $A_1 \equiv \exists\beta.B$ ,  $A_2 \equiv \exists\beta'.B$ , and  $A \equiv A_1 \sqcup A_2$ , for fresh concept names  $A_1, A_2$ ;
2. if  $A \equiv \exists(\beta \cdot \beta').B$  is a concept definition in  $\mathcal{T}_2$ , then add  $A \equiv \exists\beta.A'$  and  $A' \equiv \exists\beta'.B$  for a fresh concept name  $A'$ ;
3. if  $A \equiv \exists\beta^*.B$  is a concept definition in  $\mathcal{T}_0$ , then add  $A \equiv B \sqcup A'$  and  $A' \equiv \exists\beta.A$  for a fresh concept name  $A'$ .

The conversion clearly needs only polynomial time. The resulting TBox  $\mathcal{T}'$  is in normal form and  $\mathcal{T}$  is (finitely) satisfiable iff  $\mathcal{T}'$  is (finitely) satisfiable, as required.  $\square$

### Preliminaries for Tree Automata

We define non-deterministic (top-down) tree automata on finite trees and non-deterministic Büchi tree automata on infinite trees.

**Non-deterministic tree automata (NTA).** An NTA over finite  $k$ -ary trees is a tuple  $\mathfrak{A} = (Q, \Sigma, q_0, \Delta)$ , where  $Q$  is a set of *states*,  $q_0 \in Q$  is the *initial state*, and  $\Delta \subseteq (Q \times \Sigma) \cup (Q \times \Sigma \times Q^k)$  is the *transition relation*. The semantics is defined in terms of runs. A *run*  $r$  of  $\mathfrak{A}$  on a  $\Sigma$ -labeled tree  $(T, \tau)$  is a function  $r : T \rightarrow Q$  such that

- $r(\varepsilon) = q_0$ ,
- for every node  $x \in T$  with successors  $x_1, \dots, x_k$ , there is a transition  $(r(x), \tau(x), r(x_1), \dots, r(x_k)) \in \Delta$ , and
- for every leaf node  $x \in T$ , there is a transition  $(r(x), \tau(x)) \in \Delta$ .

The language  $L(\mathfrak{A})$  is the set of all finite  $\Sigma$ -labeled trees  $(T, \tau)$  such that there exists a run of  $\mathfrak{A}$  on  $(T, \tau)$ .

**Non-deterministic Büchi Tree Automata (NBTA).** An NBTA over  $k$ -ary trees is a tuple  $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, \Omega)$  where  $(Q, \Sigma, q_0, \Delta)$  is an NTA and  $\Omega \subseteq Q$  is a *Büchi acceptance condition*. Runs are defined as for NTAs and a run is *accepting* if every infinite path  $i_0 i_1 i_2 \dots$  in  $T$  satisfies the acceptance condition  $\Omega$ , that is, there is a state  $q \in \Omega$  such that  $r(i_j) = q$  for infinitely many  $j$ . We use  $L(\mathfrak{A})$  to denote the set of all  $\Sigma$ -labeled trees  $(T, \tau)$  such that  $\mathfrak{A}$  has an accepting run on  $(T, \tau)$ .

Note that this definition deviates slightly from the traditional definition of NBTA on  $k$ -ary trees where every node in the tree is required to have exactly  $k$  successors. In contrast, our NBTA can accept trees that have leaves and even finite trees. It is not difficult to verify that all essential properties of NBTA also hold in this model.

### Proof of Lemma 12

**Lemma 12** *Let  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$  be a 2ABTA and let  $\Sigma_{fin} = \{\sigma_1, \dots, \sigma_n\} \subseteq \Sigma$ . There is a finite set  $\mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  of existential Presburger formulas with free variables  $x_{\sigma_1}, \dots, x_{\sigma_n}$  such that for all  $\mathbf{a} \in \mathbb{N}^n$ , the following are equivalent:*

1.  $\mathbf{a}$  satisfies some  $\varphi \in \mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  over  $\mathbb{N}$ ;
2. there is a  $(T, \tau) \in L(\mathfrak{A})$  such that
  - (a)  $\mathbf{a} = \#\_{\sigma_1}(\tau), \dots, \#\_{\sigma_n}(\tau)$ ;
  - (b)  $\#\_{\sigma}(\tau) = \aleph_0$  for all  $\sigma \in \Sigma \setminus \Sigma_{fin}$ .

Moreover, there is a non-deterministic procedure that given  $\mathfrak{A}$  and  $\Sigma_{fin}$  generates exactly the formulas in  $\mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  as possible outputs, running in time single exponential in  $|Q|$  and polynomial in  $\|\delta\| + |\Sigma|$ .

**Proof.** We prove correctness of the construction given in the main part. Recall that, for some obligation pattern  $S = \{(q_1, \sigma_1, \Psi_1), \dots, (q_k, \sigma_k, \Psi_k)\}$ , the formula  $\varphi_S$  has free variables  $x_{\sigma_1}, \dots, x_{\sigma_n}$  and is defined by

$$\varphi_S = \exists \mathbf{y} \left( \varphi_G \wedge \bigwedge_{(q, \sigma) \in Z} x_{(q, \sigma)} \geq k_{q, \sigma} \right),$$

where  $\mathbf{y}$  is the tuple of variables  $x_{\sigma_{n+1}}, \dots, x_{\sigma_{n+m}}$ .

(1  $\Rightarrow$  2). Let  $\mathbf{a} \in \mathbb{N}^n$  satisfy some  $\varphi_S \in \mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$ . That is, there is some  $\mathbf{b} \in \mathbb{N}^m$  such that  $\mathbf{ab}$  satisfies  $\varphi_G$  over  $\mathbb{N}$  and also the inequalities of shape  $x_{(q, \sigma)} \leq k_{q, \sigma}$ . By definition of  $\varphi_G$ , there is a  $\Sigma \cup Z$ -labeled tree  $\tau \in L(\mathfrak{A}_S)$  such that  $\mathbf{ab} = \#\_{\sigma_1}(\tau), \dots, \#\_{\sigma_{n+m}}(\tau)$ . We construct a  $\Sigma$ -labeled tree  $\tau' \in L(\mathfrak{A})$  satisfying Conditions 2a and 2b as follows.

Since  $S$  is satisfiable, there are  $\Sigma_{inf}$ -labeled trees  $\tau_1, \dots, \tau_k$  such that for every  $i \in \{1, \dots, k\}$  we have  $\tau_i \in L(\mathfrak{A}'_{q_i})$ ,  $\tau_i(\varepsilon) = \sigma_i$ , and  $\#\_{\sigma}(\tau_i) = \aleph_0$ , for every  $\sigma \in \Psi_i$ . The idea is to obtain  $\tau'$  from  $\tau$  by replacing every leaf labeled with some  $(q_i, \sigma_i)$  with the corresponding tree  $\tau_i$ . Since the mapping is not unique, that is,  $(q_i, \sigma_i) = (q_j, \sigma_j)$  is possible even if  $i \neq j$ , there is a choice. In choosing, we have to be careful to use each tree  $\tau_i$  at least once in order to realize all symbols from  $\Sigma_{inf}$  infinitely often in the resulting tree (recall that the  $\Psi_i$  partition  $\Sigma_{inf}$ ). However, such a choice is always possible since  $\mathbf{ab}$  satisfies the inequalities in the definition of  $\varphi_S$ , that is, symbol  $(q, \sigma)$  occurs at least  $k_{q, \sigma}$  times in  $\tau$ . Thus, the resulting tree  $\tau'$  satisfies Condition 2(b). For Condition 2(a), note that the existential quantification  $\exists \mathbf{y}$  in  $\varphi_S$  essentially projects out symbols that occur infinitely often; as a result, we obtain  $\mathbf{a} = \#\_{\sigma_1}(\tau'), \dots, \#\_{\sigma_n}(\tau')$ .

(2  $\Rightarrow$  1). Assume some  $\tau \in L(\mathfrak{A})$  such that  $\mathbf{a} = \#\_{\sigma_1}(\tau), \dots, \#\_{\sigma_n}(\tau)$  and  $\#\_{\sigma}(\tau) = \aleph_0$  for every  $\sigma \in \Sigma \setminus \Sigma_{fin}$ . We show that  $\mathbf{a}$  satisfies  $\varphi_S \in \mathcal{P}_{\mathfrak{A}, \Sigma_{fin}}$  over  $\mathbb{N}$  for some satisfiable obligation pattern  $S$  for  $\mathfrak{A}'$  and  $\Sigma_{inf}$ .

Since  $\tau \in L(\mathfrak{A})$ , we can fix an accepting run  $r$  for  $\tau$ . Let  $\ell$  be a number such that below  $\ell$  no symbol from  $\Sigma_{fin}$  occurs in  $\tau$ . Since all symbols from  $\Sigma_{inf}$  occur infinitely often in  $\tau$  and  $\tau$  is finitely branching, there are nodes  $u_1, \dots, u_k$  in level  $\ell$  of the tree and a partition  $\Psi_1, \dots, \Psi_k$  of  $\Sigma_{inf}$  such that, for all  $i \in \{1, \dots, k\}$  and all  $\sigma \in \Psi_i$ , we have  $\#\_{\sigma}(\tau_i) = \aleph_0$  where  $\tau_i$  denotes the subtree of  $\tau$  rooted at  $u_i$ . Define  $S$  by taking

$$S = \{(r(u_i), \tau(u_i), \Psi_i) \mid i \in \{1, \dots, k\}\} \cup \{(r(u), \tau(u), \emptyset) \mid u \text{ in level } \ell \text{ of } \tau, u \notin \{u_1, \dots, u_k\}\}.$$

Clearly,  $S$  is a satisfiable obligation pattern for  $\mathfrak{A}'$  and  $\Sigma_{inf}$ . To show that  $\mathbf{a}$  satisfies  $\varphi_S$  over  $\mathbb{N}$ , consider the finite  $\Sigma \cup Z$ -labeled tree  $\tau'$  defined as follows:

- $\tau'$  consists of all nodes of  $\tau$  of level at most  $\ell$ ;
- $\tau'(u) = \tau(u)$ , for every node  $u \in T$  on a level smaller than  $\ell$ ;
- $\tau'(u) = (r(u), \tau(u))$ , for every node  $u \in T$  on level  $\ell$ .

By construction,  $\tau' \in L(\mathfrak{A}_S)$ . By Theorem 11, there is some  $\mathbf{b} \in \mathbb{N}^m$  such that  $\mathbf{ab}$  satisfies  $\varphi_G$  over  $\mathbb{N}$ . Since  $S$  is read off from  $\tau$  and  $r$ ,  $\mathbf{ab}$  also satisfies the inequalities of shape  $x_{(q, \sigma)} \leq k_{q, \sigma}$  in the definition of  $\varphi_S$ .  $\square$

## Proof of Lemma 13

In order to prove Lemma 13, we need an auxiliary lemma.

**Lemma 23** *For every set  $\Sigma_0 \subseteq \Sigma$ , there is a finite set  $\mathbf{A}$  of NBTAs such that:*

1. *For every  $\Sigma$ -labeled tree  $\tau$  such that  $\#\_{\sigma}(\tau) = \aleph_0$ , for all  $\sigma \in \Sigma_0$ , we have  $\tau \in L(\mathfrak{B})$  for some  $\mathfrak{B} \in \mathbf{A}$ .*
2. *For every  $\mathfrak{B} \in \mathbf{A}$  and  $\tau \in L(\mathfrak{B})$ , we have  $\#\_{\sigma}(\tau) = \aleph_0$ , for all  $\sigma \in \Sigma_0$ .*

*Moreover, there is a non-deterministic procedure that given  $\Sigma_0$  generates  $\mathbf{A}$  in polynomial time.*

**Proof.** The definition of the NBTAs in  $\mathbf{A}$  is based on the following observation.

*Claim 1.* Given a set of labels  $\Sigma' \subseteq \Sigma$ , a  $\Sigma$ -labeled tree  $(T, \tau)$ , and a node  $n \in T$ . We denote with  $T_n$  the subtree of  $T$  rooted at  $n$ . The labels in  $\Sigma'$  occur infinitely often in  $T_n$  iff one of the following is the case:

- (A) there is a node  $n' \in T_n$  and a sequence  $\Sigma_1, \dots, \Sigma_k$  of pairwise disjoint subsets of  $\Sigma'$  such that
  - $\Sigma_i \neq \Sigma'$ , for  $i \in \{1, \dots, k\}$ ,
  - $\Sigma_1 \cup \dots \cup \Sigma_k = \Sigma'$ , and
  - the labels in  $\Sigma_i$  occur infinitely often in  $T_{n'}$ ;
- (B) there is an infinite path  $n_0, n_1, \dots$  with  $n_0 = n$  such that, for every  $\sigma \in \Sigma'$ , there are infinitely many  $i$  such that some node in  $T_{n_i}$  is labeled with  $\sigma$ .

*Proof of Claim 1.* The “if”-direction is trivial. For the “only-if”-direction, assume that all labels in  $\Sigma'$  occur infinitely often in  $T_n$ , but (A) is not satisfied. We construct the infinite path in (B) inductively. Start with  $n_0 = n$ . For the induction step, let  $\Sigma_j$ ,  $j \in \{1, \dots, k\}$ , be the set of labels that occur infinitely often in the  $j$ -th successor of  $n_i$ . Clearly,  $\Sigma_1 \cup \dots \cup \Sigma_k = \Sigma'$ . Set  $\Sigma'_i = \Sigma_i \setminus (\bigcup_{j < i} \Sigma_j)$ . By definition, the  $\Sigma'_i$  are pairwise disjoint and  $\Sigma'_1 \cup \dots \cup \Sigma'_k = \Sigma'$ . Since (A) is not satisfied, for some  $j \in \{1, \dots, k\}$ , we have  $\Sigma'_j = \Sigma'$ . We set  $n_{i+1} = n_i.j$ . Consequently, for every  $i \geq 0$ , all symbols in  $\Sigma'$  occur infinitely often in  $T_{n_i}$ . Hence, Condition (B) is satisfied. This finishes the proof of the Claim.

Let now  $\Sigma_0 \subseteq \Sigma$ . Intuitively, every NBTAs in  $\mathbf{A}$  implements a sequence of partitionings of  $\Sigma_0$  according to (A), and stops partitioning at some subset  $\Sigma'$ . From then on, it continues by verifying the existence of the infinite path described in (B). It is not difficult to construct an NBTAs  $\mathfrak{B}_{\Sigma'}$  (of size polynomial in  $\Sigma'$ ) that checks Condition (B) for  $\Sigma'$ , that is,  $\mathfrak{B}_{\Sigma'}$  accepts a tree iff there is a path as described in (B); we omit the details.

Formally, the class  $\mathbf{A}$  contains an NBTAs  $\mathfrak{A}_\tau$  for every finite  $k$ -ary  $2^{\Sigma_0}$ -labeled tree  $(T, \tau)$  which satisfies the following properties:

- $\tau(\varepsilon) = \Sigma_0$ ;
- for every  $n \in T$  with successors  $n_1, \dots, n_k$ , we have that  $\tau(n_1), \dots, \tau(n_k)$  are pairwise disjoint,  $\tau(n_i) \neq \tau(n)$ , for all  $i$ , and  $\tau(n_1) \cup \dots \cup \tau(n_k) = \tau(n)$ .

Note that, by definition, the labels in the leaves of  $\tau$  form a partition of  $\Sigma_0$ . Thus, every such tree  $(T, \tau)$  has at most  $|\Sigma_0|$  leaves and thus at most  $2^{|\Sigma_0|}$  nodes.

With such a tree  $(T, \tau)$ , we associate an NBTA  $\mathfrak{A}_\tau = (Q, \Sigma, q_0, \Delta, \Omega)$  as follows. Let  $\Gamma_1, \dots, \Gamma_m$  be the labels of the leaves, and let  $\mathfrak{B}_1, \dots, \mathfrak{B}_m$  be the NBTA that check Condition (B) for  $\Gamma_1, \dots, \Gamma_m$ . Suppose  $\mathfrak{B}_i = (Q_i, \Sigma, q_{i0}, \Delta_i, \Omega_i)$ , for all  $i \in \{1, \dots, m\}$ , and denote with  $\mathfrak{B}_i^q = (Q_i^q, \Sigma, q, \Delta_i^q, \Omega_i^q)$  the variant of  $\mathfrak{B}_i$  with the initial state renamed to  $q$ . Now,  $\mathfrak{A}_\tau = (Q, \Sigma, \varepsilon, \Delta, \Omega)$  with initial state  $\varepsilon \in T$  is defined as follows:

$$Q = \{\top\} \cup T \cup \bigcup_{\substack{n \text{ leaf of } T, \\ \tau(n) = \Sigma_i}} Q_i^n$$

$$\Delta = \bigcup_{\substack{n \text{ leaf of } T, \\ \tau(n) = \Gamma_i}} \Delta_i^n \cup \{(\top, \sigma, \top, \dots, \top), (\top, \sigma)\} \cup$$

$$\bigcup_{n \text{ inner node}} \{(n, \sigma, n1, \dots, nk)\} \cup$$

$$\bigcup_{n \text{ inner node}} \{(n, \sigma, n, \top, \dots, \top), (n, \sigma, \top, n, \top, \dots, \top),$$

$$\dots, (n, \sigma, \top, \dots, \top, n)\}$$

$$\Omega = \{\top\} \cup \bigcup_{\substack{n \text{ leaf of } T, \\ \tau(n) = \Gamma_i}} \Omega_i^n$$

Thus, in states  $T$ ,  $\mathfrak{A}_\tau$  performs the partitioning according to Condition (A). In order to accept, however, it has to change at some point to some sub-automaton  $\mathfrak{B}_i$  (with the appropriate initial state) where Condition (B) is verified. More formally, we have:

*Claim 2.* The class  $\mathbf{A}$  of NBTA satisfies Conditions 1 and 2 from the Lemma.

*Proof of Claim 2.* For Condition 2, let  $\mathfrak{A}_\tau \in \mathbf{A}$  and  $\hat{\tau} \in L(\mathfrak{A}_\tau)$ , that is, there is an accepting run of  $\mathfrak{A}_\tau$  on  $\hat{\tau}$ . The Büchi acceptance condition ensures that, on every infinite path, eventually state  $\top$  or a state  $n$  for some leaf node  $n \in T$  is reached. Moreover, the transitions for inner nodes ensure that for each leaf of  $T$ , there is actually an infinite path on which state  $n$  is reached. When  $\mathfrak{A}_\tau$  visits some node  $u$  in a state  $n$ ,  $n$  leaf of  $T$ , Condition (B) for  $\tau(n)$  is verified, that is, in  $T_u$  all the symbols in  $\tau(n)$  occur infinitely often. Since the labels in the leaves of  $\tau$  partition  $\Sigma_0$ , all symbols in  $\Sigma_0$  occur infinitely often in  $\hat{\tau}$ .

For Condition 1, let  $\tau$  be a  $\Sigma$ -labeled tree such that  $\#_\sigma(\tau) = \aleph_0$ , for every  $\sigma \in \Sigma_0$ . We construct a  $2^{\Sigma_0}$ -labeled tree  $\hat{\tau}$  and an accepting run  $r$  of  $\mathfrak{A}_\tau$  on  $\hat{\tau}$  as follows. Start with a single node tree  $\hat{\tau}$  with  $\hat{\tau}(\varepsilon) = \Sigma_0$  and  $r(\varepsilon) = \varepsilon$ . Then apply the following rule exhaustively:

- Choose a leaf node  $n$  in  $\hat{\tau}$  and let  $\hat{\tau}(r(n)) = \Sigma'$ .
  - If  $\Sigma'$  and  $n$  satisfy Condition (A) above, let  $n'$  be the witnessing node in  $T_n$  and let  $\Sigma_1, \dots, \Sigma_k$  be the witnessing partition of  $\Sigma'$ . Set  $r(m) = n$  for every  $m$  on the path from  $n$  to  $n'$ , and  $r(m) = \top$  for every  $m$  that is a successor node of some node on the path from  $n$  to

$n'$  except  $n'$ , but not on the path itself. Moreover, add  $k$  successors to  $n$  in  $\hat{\tau}$ , set  $\hat{\tau}(ni) = \Sigma_i$  and  $r(n'i) = ni$  for all  $i \in \{1, \dots, k\}$ .

- Otherwise, by Claim 1,  $\Sigma'$  and  $n$  satisfy Condition (B) above. By definition, the NBTA  $\mathfrak{B}_\Sigma^n$  has an accepting run  $r'$  on the subtree  $T_n$  rooted at  $n$ . Then complete the run  $r$  by setting  $r(n') = r'(n')$ , for every  $n' \in T_n$ ,  $n \neq n'$ .

It is routine to verify that the constructed  $\hat{\tau}$  is a finite,  $k$ -ary,  $2^{\Sigma_0}$ -labeled tree with the required properties. Moreover, the resulting run  $r$  is an accepting run of  $\mathfrak{A}_\tau$  on  $\hat{\tau}$ . This finishes the proof of the Claim.

It remains to argue for the ‘moreover’ part of the lemma. Given  $\Sigma_0$ , we can guess a tree  $(T, \tau)$  and construct the corresponding automaton  $\mathfrak{A}_\tau$ . As mentioned above, the tree is of polynomial size, and thus also the automaton is.  $\square$

Now we can finish the proof of Lemma 13, which we restate for the sake of convenience.

**Lemma 13** *Given an NBTA  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ , a set  $\Sigma_{inf} \subseteq \Sigma$ , and an obligation  $(q, \sigma, \Psi)$  for  $\mathfrak{A}$  and  $\Sigma_{inf}$ , it is decidable in NP whether  $(q, \sigma, \Psi)$  is satisfiable.*

**Proof.** Let  $(q, \sigma, \Psi)$  be an obligation for  $\mathfrak{A}$  and  $\Sigma_{inf}$ , and let further  $\mathbf{A}$  be the set of NBTA from Lemma 23 applied to the case with  $\Sigma = \Sigma_{inf}$  and  $\Sigma_0 = \Psi$ . Then  $(q, \sigma, \Psi)$  is satisfiable iff there is some  $\mathfrak{B} \in \mathbf{A}$  such that the intersection of  $\mathfrak{A}_q$  and  $\mathfrak{B}$  is non-empty. This is an NP algorithm since according to the ‘moreover’ part of Lemma 23,  $\mathfrak{B}$  can be guessed in polynomial time, and intersection non-emptiness of NBTA is possible in polynomial time.  $\square$

## Proof of Proposition 15

**Proposition 15** *An  $ALCCOL\mathcal{F}_{reg}^1$ -TBox  $\mathcal{T}$  in normal form is (finitely) satisfiable iff there is a set  $\Gamma \subseteq \text{tp}(\mathcal{T})$  such that*

1. *there is a model  $\mathcal{I}_{reg}$  of  $\mathcal{T}_{bool} \cup \mathcal{T}_{reg}$  that realizes  $\Gamma$ ;*
2. *there are (finite) models  $\mathcal{I}_r$  of  $\mathcal{T}_{bool} \cup \mathcal{T}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , such that for all  $r, s \in \text{Fn}(\mathcal{T})$ , the following conditions hold:*
  - (a)  $\mathcal{I}_r$  realizes  $\Gamma$ ;
  - (b)  $\#_t(\mathcal{I}_r) = \#_t(\mathcal{I}_s)$  for all  $t \in \Gamma$ .

**Proof.** ( $\Rightarrow$ ) Let  $\mathcal{I}$  be a (finite) model of  $\mathcal{T}$ , and let  $\Gamma$  be the set of types realized in  $\mathcal{I}$ . Then,  $\mathcal{I}_{reg} = \mathcal{I}$  witnesses Condition 1 and  $\mathcal{I}_r = \mathcal{I}$ ,  $r \in \text{Fn}(\mathcal{T})$ , witness Condition 2.

( $\Leftarrow$ ) Let  $\Gamma$  be a set of types for  $\mathcal{T}$  that satisfies Conditions 1 and 2, that is, there is a model  $\mathcal{I}_{reg}$  of  $\mathcal{T}_{reg}$  that realizes  $\Gamma$  and models  $\mathcal{I}_r$  of  $\mathcal{T}_{reg} \cup \mathcal{T}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ , which satisfy Conditions 2(a)–(b).

We construct a model  $\mathcal{I}$  of  $\mathcal{T}$  as follows. If there is no functionality assertion in  $\mathcal{T}$ , take  $\mathcal{I} = \mathcal{I}_{reg}$  and we are done. Otherwise, fix some  $r \in \text{Fn}(\mathcal{T})$ . Condition 2(b) implies that we can choose bijections  $\pi_s : \Delta^{\mathcal{I}_r} \rightarrow \Delta^{\mathcal{I}_s}$  for every  $s \in \text{Fn}(\mathcal{T})$  such that  $\text{tp}_{\mathcal{I}_r}(d) = \text{tp}_{\mathcal{I}_s}(\pi_s(d))$  for all  $d \in \Delta^{\mathcal{I}_r}$ . For types  $t, t' \in \Gamma$  and a role name  $s$ , we write  $t \rightsquigarrow_s^{\mathcal{T}} t'$  if

1. for all CIs  $A \equiv \exists s. B \in \mathcal{T}$ ,  $B \in t'$  implies  $A \in t$ ;
2. for all CIs  $A \equiv \exists s^-. B \in \mathcal{T}$ ,  $B \in t$  implies  $A \in t'$ .

The interpretation  $\mathcal{I}$  is now defined as follows:

$$\begin{aligned}\Delta^{\mathcal{I}} &= \Delta^{\mathcal{I}_r} \\ A^{\mathcal{I}} &= A^{\mathcal{I}_r} \\ a^{\mathcal{I}} &= a^{\mathcal{I}_r} \\ s^{\mathcal{I}} &= \{(d, e) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (\pi(d), \pi(e)) \in s^{\mathcal{I}_s}\} \\ s_0^{\mathcal{I}} &= \{(d, e) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \text{tp}_{\mathcal{I}}(d) \rightsquigarrow_{s_0}^{\mathcal{I}} \text{tp}_{\mathcal{I}}(e)\}\end{aligned}$$

for all  $A \in \mathbb{N}_{\mathcal{C}}$ ,  $a \in \mathbb{N}_{\mathcal{I}}$ ,  $s \in \text{Fn}(\mathcal{T})$ , and  $s_0 \in \mathbb{N}_{\mathbb{R}} \setminus \text{Fn}(\mathcal{T})$ .

To show that  $\mathcal{I} \models \mathcal{T}$ , it suffices to show that  $\mathcal{I} \models \mathcal{T}_{\text{bool}}$ ,  $\mathcal{I} \models \mathcal{T}_{\text{reg}}$ , and  $\mathcal{I} \models \mathcal{T}_s$  for all  $s \in \text{Fn}(\mathcal{T})$ . By construction of  $\mathcal{I}$ , the restrictions of  $\mathcal{I}$  and  $\mathcal{I}_s$  to role name  $s$  are isomorphic (witnessed by  $\pi_s$ ). Since each  $\mathcal{I}_s$  is a model of  $\mathcal{T}_{\text{bool}} \cup \mathcal{T}_s$ , we obtain that  $\mathcal{I} \models \mathcal{T}_{\text{bool}} \cup \mathcal{T}_s$  for all  $s \in \text{Fn}(\mathcal{T})$ . It remains to show that  $\mathcal{I} \models \mathcal{T}_{\text{reg}}$ . Let  $A \equiv \exists \alpha. B \in \mathcal{T}_{\text{reg}}$ . We show the two directions of the equivalence  $A^{\mathcal{I}} = (\exists \alpha. B)^{\mathcal{I}}$  separately.

For “ $\subseteq$ ”, one first proves the following claim by induction on the structure of  $\alpha$ .

*Claim 1.* For all roles  $\alpha$  in  $\mathcal{T}_{\text{reg}}$ , all  $d, e \in \Delta^{\mathcal{I}}$ , and all  $d', e' \in \Delta^{\mathcal{I}_{\text{reg}}}$  such that  $\text{tp}_{\mathcal{I}}(d) = \text{tp}_{\mathcal{I}_{\text{reg}}}(d')$  and  $\text{tp}_{\mathcal{I}}(e) = \text{tp}_{\mathcal{I}_{\text{reg}}}(e')$ ,  $(d', e') \in \alpha^{\mathcal{I}_{\text{reg}}}$  implies  $(d, e) \in \alpha^{\mathcal{I}}$ .

*Proof of Claim 1.* The induction base follows from the fact that  $\text{tp}_{\mathcal{I}_{\text{reg}}}(d) \rightsquigarrow_s \text{tp}_{\mathcal{I}_{\text{reg}}}(e)$  for all  $(d, e) \in s^{\mathcal{I}_{\text{reg}}}$ . The induction step is then immediate.

Now let  $d \in A^{\mathcal{I}}$ . Since  $\mathcal{I}_{\text{reg}}$  and  $\mathcal{I}$  both realize  $\Gamma$ , there is an element  $d' \in \Delta^{\mathcal{I}_{\text{reg}}}$  such that  $\text{tp}_{\mathcal{I}}(d) = \text{tp}_{\mathcal{I}_{\text{reg}}}(d')$ . Since  $\mathcal{I}_{\text{reg}} \models \mathcal{T}_{\text{reg}}$ , there is some  $e'$  such that  $(d', e') \in \alpha^{\mathcal{I}_{\text{reg}}}$  and  $e' \in B^{\mathcal{I}_{\text{reg}}}$ . Since  $\mathcal{I}_{\text{reg}}$  and  $\mathcal{I}$  both realize  $\Gamma$ , there is an element  $e$  with  $\text{tp}_{\mathcal{I}}(e) = \text{tp}_{\mathcal{I}_{\text{reg}}}(e')$ . Claim 1 yields  $d \in (\exists \alpha. B)^{\mathcal{I}}$ .

For the “ $\supseteq$ ” direction, we prove by induction on  $\alpha$  that for all  $A \equiv \exists \alpha. B \in \mathcal{T}_{\text{reg}}$ ,  $d \in (\exists \alpha. B)^{\mathcal{I}}$  implies  $d \in A^{\mathcal{I}}$ :

- If  $\alpha$  is a test  $X?$ , then  $X$  is a concept name,  $d \in X^{\mathcal{I}}$ , and  $d \in B^{\mathcal{I}}$ , and hence  $d \in X^{\mathcal{I}_r}$  and  $d \in B^{\mathcal{I}_r}$ . Since  $A \equiv \exists X?.B \in \mathcal{T}_r$ , and  $\mathcal{I}_r \models \mathcal{T}_r$ , we have  $d \in A^{\mathcal{I}_r}$  and thus  $d \in A^{\mathcal{I}}$ .
- If  $\alpha = s$  or  $\alpha = s^-$ , for some role name  $s$ , then the statement follows from the definition of  $\rightsquigarrow_s^{\mathcal{I}}$ .
- If  $\alpha = \beta \cdot \beta'$ , then there is some  $e \in \Delta^{\mathcal{I}}$  such that  $(d, e) \in \beta^{\mathcal{I}}$  and  $e \in (\exists \beta'. B)^{\mathcal{I}}$ . By Condition 6 of the normal form,  $\mathcal{T}_{\text{reg}}$  contains concept definitions  $A \equiv \exists \beta. A_1$  and  $A_1 \equiv \exists \beta'. B$ . By induction, we have  $e \in A_1^{\mathcal{I}}$ , and, again by induction, also  $d \in A^{\mathcal{I}}$ .
- If  $\alpha = \beta + \beta'$ , then  $d \in (\exists \beta. B)^{\mathcal{I}}$  or  $d \in (\exists \beta'. B)^{\mathcal{I}}$ . By Condition 5 of the normal form,  $\mathcal{T}_{\text{reg}}$  contains concept definitions  $A_1 \equiv \exists \beta. B$ ,  $A_2 \equiv \exists \beta'. B$ , and  $A \equiv A_1 \sqcup A_2$ . By induction, we have  $d \in A_1^{\mathcal{I}}$  or  $d \in A_2^{\mathcal{I}}$ , and thus  $d \in A^{\mathcal{I}}$ .
- If  $\alpha = \beta^*$ , there are elements  $d_0, \dots, d_n, n \geq 0$ , such that  $d_0 = d$ ,  $d_n \in B^{\mathcal{I}}$ , and  $(d_i, d_{i+1}) \in \beta^{\mathcal{I}}$ , for all  $i$  with  $0 \leq i < n$ . By Condition 7 of the normal form,  $\mathcal{T}$  contains concept definitions  $A_1 \equiv \exists \beta. A$  and  $A \equiv B \sqcup A_1$ . We show inductively that  $d_i \in A^{\mathcal{I}}$  for all  $i$  with  $0 \leq i \leq n$ .

Since  $d_n \in B^{\mathcal{I}}$  and  $A \equiv B \sqcup A_1 \in \mathcal{T}$ , we obtain  $d_n \in A^{\mathcal{I}}$ . For  $i < n$ , we know  $d_{i+1} \in A^{\mathcal{I}}$  by induction. Since  $(d_i, d_{i+1}) \in \beta^{\mathcal{I}}$  and  $A_1 \equiv \exists \beta. A$ , we obtain  $d_i \in A_1^{\mathcal{I}}$ . Since  $A \equiv B \sqcup A_1 \in \mathcal{T}$ , we obtain  $d_i \in A^{\mathcal{I}}$ .  $\square$

## Proof of Proposition 16

Proposition 16 is an immediate consequence of Lemma 17, so we will only give the proof of the latter.

For a finite automaton  $\mathfrak{B} = (S, \Omega, s_0, \Delta, F)$ , and any  $s \in S$ , we use  $\mathfrak{B}_s$  to denote the automaton obtained from  $\mathfrak{B}$  by replacing the initial state with  $s$ . Moreover, given a regular role  $\alpha$ , we denote with  $\mathfrak{B}_\alpha$  the finite automaton equivalent to  $\alpha$  which exists according to Lemma 22.

**Lemma 17** *One can construct in time single exponential in  $\|\mathcal{T}\|$  a 2ABTA $_{\exists PP}$   $\mathfrak{A}_{\mathcal{T}}$  whose number of states is polynomial in  $\|\mathcal{T}\|$  and such that  $L(\mathfrak{A}_{\mathcal{T}})$  consists of exactly those binary well-formed  $\Sigma$ -labeled trees  $(T, \tau)$  such that for  $\mathcal{I}_r = \mathcal{I}_r^r$ ,  $r \in \text{Fn}(\mathcal{T})$ , Condition 2 of Proposition 15 is satisfied.*

**Proof.** The 2ABTA $_{\exists PP}$   $\mathfrak{A}_1, \mathfrak{A}_3$  are straightforward to construct; we omit details. For  $\mathfrak{A}_2 = (Q_2, \Sigma, q_{0,2}, \delta_2, \Omega_2, \mathcal{C}_2)$ , we use only the set of Presburger constraints  $\mathcal{C}_2$  (and leave the actual automaton empty), which contains the following equations, for every  $t \in \Gamma$ :

$$\begin{aligned}\sum_{(R,t,L) \in \Sigma} x_{(R,t,L)} &> 0 \tag{1} \\ \sum_{\substack{(R,t,L) \in \Sigma, \\ R \in \{r, r^-\}}} x_{(R,t,L)} &= \sum_{\substack{(S,t,L) \in \Sigma, \\ S \in \{s, s^-\}}} x_{(S,t,L)} \quad \text{for } r, s \in \text{Fn}(\mathcal{T}). \tag{2}\end{aligned}$$

Intuitively, Equation (2) establishes the synchronization of the type multiplicities.

Now, for the 2ABTA $_{\exists PP}$   $\mathfrak{A}_r$ ,  $r \in \text{Fn}(\mathcal{T})$ . We fix one  $r \in \text{Fn}(\mathcal{T})$  throughout the rest of the proof and define  $\mathfrak{A}_r = (Q, \Sigma, q_0, \delta, \Omega, \mathcal{C})$  with  $\mathcal{C} = \emptyset$ . The set  $Q$  consists of  $q_0, q^*$ , and all states of the form

$$q_{\mathfrak{B}_s, B}, q_{\mathfrak{B}_s, B}^a, q_{\mathfrak{B}_s, B}^{a, X}, \bar{q}_{\mathfrak{B}_s, B}, \bar{q}_{\mathfrak{B}_s, B}^a, \bar{q}_{\mathfrak{B}_s, B}^{a, X}$$

where  $\mathfrak{B}$  is a finite automaton equivalent to some regular expression  $\alpha$  that occurs in  $\mathcal{T}_r$  (c.f. Lemma 22),  $s$  is a state in  $\mathfrak{B}$ ,  $B$  is a concept name that occurs in  $\mathcal{T}$ ,  $a$  is an alphabet symbol in  $\mathfrak{B}$  and  $X \in \{\uparrow, \downarrow, \text{src}, \text{tgt}\}$ . The Büchi condition  $\Omega$  consists of all states of the form  $\bar{q}_{\mathfrak{B}_s, B}^*$  (which will be explained below).

We next define the transition function  $\delta$ . As a general proviso, we provide the definition of  $\delta(q, \sigma)$  only for the ‘relevant’ inputs  $q, \sigma$ , that is, inputs which can be reached during a successful run of the automaton on a well-formed tree. For all other  $q, \sigma$ , we implicitly set  $\delta(q, \sigma) = \text{false}$ .

The automaton visits every node in the input tree and verifies that it satisfies the concept definitions in  $\mathcal{T}_{\text{bool}} \cup \mathcal{T}_r$ . This process is initiated using the following transitions.

$$\begin{aligned}\delta(q_0, \circ) &= \square q_0 \\ \delta(q_0, \sigma) &= q^* \wedge \square q_0 \quad \text{for all } \sigma \neq \circ\end{aligned}$$

When the automaton visits a node  $n$  in state  $q^*$  this represents the obligation to verify that (the domain element represented by)  $n$  satisfies the concept definitions in  $\mathcal{T}_r$ . We include the following transitions for all  $(R, t, L) \in \Sigma$  with  $R \in \{r, r^-\}$  and  $(S, t, L) \in \Sigma$  with  $S \notin \{r, r^-\}$ :

$$\delta(q^*, (S, t, L)) = \text{true}$$

$$\delta(q^*, (R, t, L)) = \bigwedge_{A \in t, A \equiv \exists \alpha. B \in \mathcal{T}_r} q_{\mathfrak{B}, \alpha, B} \wedge \bigwedge_{A \notin t, A \equiv \exists \alpha. B \in \mathcal{T}_r} \bar{q}_{\mathfrak{B}, \alpha, B}$$

When the automaton visits a node  $n$  in a state  $q_{\mathfrak{B}, B}$ , it has the obligation to find a node  $n' \in B^{\mathcal{I}_r}$  with  $(n, n') \in E_{\mathfrak{B}}^{\mathcal{I}_r}$ . Complementary, the visit of  $n$  in state  $\bar{q}_{\mathfrak{B}, B}$  represents the obligation to verify that all nodes  $n'$  with  $(n, n') \in E_{\mathfrak{B}}^{\mathcal{I}_r}$  do not satisfy  $n' \in B^{\mathcal{I}_r}$ . We include the following transitions for states  $q_{\mathfrak{B}, B}$  with  $\mathfrak{B} = (S, \Omega, s_0, \Delta, F)$ :

$$\delta(q_{\mathfrak{B}, B}, (R, t, L)) = \text{true} \quad \text{if } s_0 \in F \text{ and } B \in t$$

$$\delta(q_{\mathfrak{B}, B}, (R, t, L)) = \bigvee_{(s_0, a, s) \in \Delta} q_{\mathfrak{B}, s, B}^a \quad \text{if } s_0 \notin F \text{ or } B \notin t$$

In a state  $q_{\mathfrak{B}, B}^a$ , the automaton has the obligation to simulate the step  $a$  in the represented interpretation and change to state  $q_{\mathfrak{B}, B}$ . This is easily realized for tests  $A^?$ :

$$\delta(q_{\mathfrak{B}, B}^{A^?}, (R, t, L)) = q_{\mathfrak{B}, B} \quad \text{if } A \in t$$

$$\delta(q_{\mathfrak{B}, B}^{A^?}, (R, t, L)) = \text{false} \quad \text{if } A \notin t$$

For the case when  $a$  is a role  $R$ , the automaton has to navigate to an  $R$  successor of the current element, taking into account the encoding of structures as  $\Sigma$ -labeled trees. In this part, the automaton uses the states of the form  $q_{\mathfrak{B}, B}^{R, X}$  with  $R \in \{r, r^-\}$  and  $X \in \{\uparrow, \downarrow, \text{src}, \text{tgt}\}$ . We concentrate on the case when  $R = r$  is a role name; the case for inverse roles  $r^-$  is symmetric and can be obtained from the given case by replacing  $r$  with  $r^-$  and additionally exchanging the role of  $\text{src}$  and  $\text{tgt}$ . Before we give the actual transitions, we provide a summary of the intended behavior of the used states when visiting a node  $n$ .

- $q_{\mathfrak{B}, B}^{r, \uparrow}$ : navigate to  $n^\uparrow$ ;
- $q_{\mathfrak{B}, B}^{r, \downarrow}$ : navigate to some  $n'$  such that  $n = (n')^\uparrow$ ;
- $q_{\mathfrak{B}, B}^{r, \text{src}}$ : check if  $n$  is labeled with  $\text{src}$ ; if so, change to  $q_{\mathfrak{B}, B}^{r, \text{tgt}}$ ;
- $q_{\mathfrak{B}, B}^{r, \text{tgt}}$ : navigate to the corresponding node with  $\text{tgt} \in L$ .

Formally, we use the following transitions:

$$\delta(q_{\mathfrak{B}, B}^{r, \uparrow}, (S, t, L)) = \text{false} \quad \text{if } S \notin \{r, r^-\}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \uparrow}, (r^-, t, L)) = \diamond^- q_{\mathfrak{B}, B}^{r, \uparrow} \vee \diamond q_{\mathfrak{B}, B}^{r, \downarrow} \vee q_{\mathfrak{B}, B}^{r, \text{src}}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \downarrow}, (r, t, L)) = \diamond q_{\mathfrak{B}, B}^{r, \downarrow} \vee q_{\mathfrak{B}, B}^{r, \text{src}}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{src}}, (r, t, L)) = \text{false} \quad \text{if } \text{src} \notin L$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{src}}, (r, t, L)) = q_{\mathfrak{B}, B}^{r, \text{tgt}} \quad \text{if } \text{src} \in L$$

$$\delta(q_{\mathfrak{B}, B}^{r, \uparrow}, \circ) = \diamond^- q_{\mathfrak{B}, B}^{r, \uparrow}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \uparrow}, (S, t, L)) = q_{\mathfrak{B}, B}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \downarrow}, \circ) = \diamond q_{\mathfrak{B}, B}^{r, \downarrow}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \downarrow}, (r, t, L)) = q_{\mathfrak{B}, B}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \downarrow}, (r^-, t, L)) = \text{false}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{tgt}}, (S, t, L)) = \text{false} \quad \text{if } S \notin \{r, r^-\}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{tgt}}, (r, t, L)) = q_{\mathfrak{B}, B} \quad \text{if } \text{tgt} \in L$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{tgt}}, (r^-, t, L)) = q_{\mathfrak{B}, B} \quad \text{if } \text{tgt} \in L$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{tgt}}, \circ) = \diamond^- q_{\mathfrak{B}, B}^{r, \text{tgt}} \vee \diamond q_{\mathfrak{B}, B}^{r, \text{tgt}}$$

$$\delta(q_{\mathfrak{B}, B}^{r, \text{tgt}}, (S, t, L)) = \diamond^- q_{\mathfrak{B}, B}^{r, \text{tgt}} \vee \diamond q_{\mathfrak{B}, B}^{r, \text{tgt}}$$

$$\text{if } S \in \{r, r^-\} \text{ and } \text{tgt} \notin L$$

The transitions for states of the form  $\bar{q}_*$  are complementary, that is, they can be obtained from the transitions given above by replacing false by true, true by false,  $\vee$  by  $\wedge$ ,  $\diamond$  by  $\square$ ,  $\diamond^-$  by  $\square^-$ , and all states  $q_*^*$  with  $\bar{q}_*$ .

The Büchi condition ensures that the automaton cannot stay in states of the form  $q_{\mathfrak{B}, B}^*$  along an infinite path of a run. Thus, existential restrictions  $\exists \alpha. B$  are eventually witnessed.

Based on the provided intuitions, it is routine to verify that the constructed automaton  $\mathfrak{A}_r$  accepts a well-formed  $(T, \tau)$  iff  $\mathcal{I}_r$  is a model of  $\mathcal{T}_{\text{bool}} \cup \mathcal{T}_r$ . It remains to note that  $\mathfrak{A}_r$  has polynomially many states and can be constructed in single exponential time.

Overall, the intersection  $\mathfrak{A}_{\mathcal{T}}$  of  $\mathfrak{A}_1, \mathfrak{A}_2, \mathfrak{A}_3$  and  $\mathfrak{A}_r$ ,  $r \in \text{Fn}(\mathcal{T})$  has polynomially many states and can be constructed in single exponential time by Lemma 8.  $\square$

## Proof of Proposition 19

Proposition 19 is an immediate consequence of Lemma 20, so we will give only the proof of the latter. We start with spelling out the additional progress condition on types mentioned in the main body of the paper.

A *type for  $\mathcal{T}$*  is a set  $t$  of concept names used in  $\mathcal{T}$  such that there is a model  $\mathcal{I}$  of  $\mathcal{T}_{\text{bool}}$  and a  $d \in \Delta^{\mathcal{I}}$  such that  $t = \{A \text{ used in } \mathcal{T} \mid d \in A^{\mathcal{I}}\}$  and additionally, the following progress condition is satisfied:

(P) if  $A \in t$  and  $A \equiv \exists \alpha. B \in \mathcal{T}$ , then one of the following is the case:

1.  $B \in t$  and there are  $A_1, \dots, A_n \in t$  such that  $A_1? \dots A_n? \in L(\alpha)$ ;
2. there are concept names  $A_1, \dots, A_n$  and  $B_0, \dots, B_m$  with  $B_m = B$ , a role  $R$ , and regular roles  $\beta_1, \dots, \beta_{m-1}$  such that:

$$(a) L(A_1? \dots A_n? R \beta_1 \dots \beta_{m-1}) \subseteq L(\alpha);$$

$$(b) B_0 \equiv \exists R. B_1 \in \mathcal{T};$$

$$(c) B_i \equiv \exists \beta_i. B_{i+1} \in \mathcal{T} \text{ for } 1 \leq i < m;$$

$$(d) A_1, \dots, A_n, B_0 \in t.$$

We give a simple example to illustrate the need for this condition. Let  $A \in t$ ,  $B \notin t$ , and  $A \equiv \exists (A_0? + r)^*. B \in$



$\mathcal{T}$ . Then together with Conditions 5-8 of TBoxes in normal form, (P) makes sure that we find some  $X \in t$  with  $X \equiv \exists r.A \in \mathcal{T}$ . Note that Conditions 5-8 alone would allow us to run into a ‘bad cycle’, endlessly deferring the satisfaction of  $\exists(A_0? + r).B$  by running through the test  $A_0?$  over and over again. The same can of course happen in  $\mathcal{ALCOIF}_{reg}^1$ , where we did not impose this extra condition. In the decision procedure used there, such ‘local bad cycles’ were ruled out by the 2ABTA in a uniform way along with ‘non-local bad cycles’. This is not easily possible in our decision procedure for  $\mathcal{ALCOIF}_{reg}^2$  because of the ‘slice’ representation of interpretations in  $\Sigma$ -labeled trees.

The following lemma asserts soundness of (P).

**Lemma 24** *Let  $\mathcal{I}$  be a model of  $\mathcal{T}$  and  $d \in \Delta^{\mathcal{I}}$ . Then the set  $t = \{A \text{ used in } \mathcal{T} \mid d \in A^{\mathcal{I}}\}$  satisfies (P).*

**Proof.** The lemma is an immediate consequence of the following Claim that also provides a bound on the length of the sequences.

*Claim.* Let  $\mathcal{I}$  be a model of  $\mathcal{T}$ . For all  $A \equiv \exists\alpha.B \in \mathcal{T}$ , and all  $d, e \in \Delta^{\mathcal{I}}$  with  $d \in A^{\mathcal{I}}$ ,  $e \in B^{\mathcal{I}}$ , and  $(d, e) \in \alpha^{\mathcal{I}}$ , one of the following is the case:

1.  $d = e$  and there are concept names  $A_1, \dots, A_n$  such that  $n \leq \|\alpha\|$ ,  $A_1? \dots A_n? \in L(\alpha)$ , and  $d \in A_i^{\mathcal{I}}$ , for all  $i \in \{1, \dots, n\}$ ;
2. there are concept names  $A_1, \dots, A_n$  and  $B_0, \dots, B_m$  with  $B_m = B$ , a role  $R$ , and regular roles  $\beta_1 \dots, \beta_{m-1}$  such that:
  - (a)  $L(A_1? \dots A_n? R \beta_1 \dots \beta_{m-1}) \subseteq L(\alpha)$ ;
  - (b)  $B_0 \equiv \exists R.B_1 \in \mathcal{T}$ ;
  - (c)  $B_i \equiv \exists \beta_i.B_{i+1} \in \mathcal{T}$ , for  $1 \leq i < m$ ;
  - (d)  $d \in B_0^{\mathcal{I}}$  and  $d \in A_i^{\mathcal{I}}$ , for  $1 \leq i \leq n$ ;
  - (e)  $(d, e) \in (A_1? \dots A_n? R \beta_1 \dots \beta_{m-1})^{\mathcal{I}}$ ;
  - (f)  $n + m \leq \|\alpha\|$ .

Note that Conditions 2(a)–(d) in the Claim correspond exactly to the Conditions 2(a)–(d) of Condition (P). Condition 2(e) is needed to strengthen the induction hypothesis; Condition 2(f) establishes the announced bound. The proof of the Claim is by induction over the structure of the regular role  $\alpha$  and heavily relies on the TBox normal form. The stronger induction hypothesis in 2(f) is needed in the cases for composition  $\alpha_1 \cdot \alpha_2$  and Kleene-star  $\beta^*$ .

If  $\alpha = X?$ , then by the TBox normal form, we have  $A \equiv X \sqcap B \in \mathcal{T}$ . Since  $\mathcal{I}$  is a model of  $\mathcal{T}$  and  $d \in A^{\mathcal{I}}$ , we also have  $d \in X^{\mathcal{I}}$  and  $d \in B^{\mathcal{I}}$ . Thus, we are in Case 1.

If  $\alpha = S$  is a role, then setting  $n = 0$ ,  $R = S$ ,  $B_0 = A$ ,  $m = 1$ , and  $B_1 = B$  witnesses that we are in Case 2.

If  $\alpha = \alpha_1 + \alpha_2$ , then by the TBox normal form there are concept names  $X_1, X_2$  such that  $A \equiv X_1 \sqcup X_2 \in \mathcal{T}$  and  $X_i \equiv \exists\alpha_i.B \in \mathcal{T}$ . Since  $\mathcal{I}$  is a model of  $\mathcal{T}$ , we have  $d \in X_i^{\mathcal{I}}$  and  $(d, e) \in \alpha_i$ , for some  $i \in \{1, 2\}$ . By induction applied to  $X_i \equiv \exists\alpha_i.B \in \mathcal{T}$  and  $d, e$ , either Case 1 or Case 2 is true for  $X_i \equiv \exists\alpha_i.B \in \mathcal{T}$  and  $d, e$ . Obviously, the same is true for  $A \equiv \exists\alpha.B$  and  $d, e$ .

If  $\alpha = \alpha_1 \cdot \alpha_2$ , then by the TBox normal form there is a concept name  $X$  such that  $A \equiv \exists\alpha_1.X \in \mathcal{T}$  and

$X \equiv \exists\alpha_2.B \in \mathcal{T}$ . Since  $\mathcal{I}$  is a model of  $\mathcal{T}$ , there is some  $e'$  such that  $(d, e') \in \alpha_1^{\mathcal{I}}$  and  $e' \in X^{\mathcal{I}}$ . Applying the induction hypothesis to  $A \equiv \alpha_1.X \in \mathcal{T}$  and  $d, e'$  leaves us with two cases:

- There are  $A_1, \dots, A_n$  such that  $A_1? \dots A_n? \in L(\alpha_1)$ ,  $d = e'$ , and  $d \in A_i^{\mathcal{I}}$ , for all  $i \in \{1, \dots, n\}$ . Applying the induction hypothesis also to  $X \equiv \alpha_2.B \in \mathcal{T}$  and  $(e', e)$  yields again two cases:
  - There are  $B_1, \dots, B_k$  with  $D_1? \dots B_k? \in L(\alpha_2)$ ,  $e' = e$ , and  $e' \in B_i^{\mathcal{I}}$ , for all  $i \in \{1, \dots, k\}$ . Then the sequence  $A_1, \dots, A_n, B_1, \dots, B_k$  witnesses that Case 1 is true for  $A \equiv \exists\alpha.B \in \mathcal{T}$  and  $d, e$ .
  - There are concept names  $A'_1, \dots, A'_k$  and  $B_0, \dots, B_m$  with  $B_m = B$ , a role  $R$ , and regular roles  $\beta_1 \dots, \beta_{m-1}$  satisfying Condition (a)–(f) from Case 2. Now, the sequences  $A_1, \dots, A_n, A'_1, \dots, A'_k$  and  $B_1, \dots, B_m$ , the role  $R$  and the regular roles  $\beta_1, \dots, \beta_{m-1}$  witness that Case 2 is true for  $A \equiv \exists\alpha.B$  and  $d, e$ .
- There are concept names  $A_1, \dots, A_n \in t$  and  $B_0, \dots, B_m$  with  $B_m = X$ , a role  $R$ , and regular roles  $\beta_1 \dots, \beta_{m-1}$  such that Conditions (a)–(f) from Case 2 are satisfied. Define  $B_{m+1} = B$  and  $\beta_m = \alpha_2$ . Obviously, the sequences  $A_1, \dots, A_n$  and  $B_1, \dots, B_{m+1}$ , the role  $R$  and the sequence  $\beta_1, \dots, \beta_m$  of regular roles witness that Case 2 is true for  $A \equiv \exists\alpha.B$  and  $d, e$ .

If  $\alpha = \beta^*$ , then by the TBox normal form there is a concept name  $X$  such that  $A \equiv B \sqcup X \in \mathcal{T}$  and  $X \equiv \exists\beta.A \in \mathcal{T}$ . Since  $\mathcal{I}$  is a model of  $\mathcal{T}$ , we have either  $d \in B^{\mathcal{I}}$  or  $d \in X^{\mathcal{I}}$ . In the former case, setting  $n = 0$  witnesses that we are in Case 1. In the latter case, there is a sequence of elements  $d_0, \dots, d_k$ ,  $k > 0$  such that  $d_0 = d$ ,  $d_n = e$ ,  $(d_i, d_{i+1}) \in \beta^{\mathcal{I}}$ , for all  $i \in \{0, \dots, k-1\}$ , and  $d_n \in B^{\mathcal{I}}$ . We can assume that  $d_0 \neq d_1$ . To see why this is without loss of generality, let  $j$  be the smallest  $\ell$  with  $d_\ell \neq d_0$ . If there is no such  $\ell$ ,  $e = d_k = d_0 = d$ , and we are in the first case. Otherwise, drop  $d_1, \dots, d_{j-1}$  from the sequence.

We apply induction to  $X \equiv \exists\beta.A \in \mathcal{T}$ ,  $d_0 \in X^{\mathcal{I}}$ ,  $d_1 \in A^{\mathcal{I}}$ , and  $(d_0, d_1) \in \beta^{\mathcal{I}}$ . As  $d_0 \neq d_1$ , Case 2 has to be satisfied, that is, there are concept names  $A_1, \dots, A_n$  and  $B_0, \dots, B_m$  with  $B_m = A$ , a role  $R$ , and regular roles  $\beta_1 \dots, \beta_{m-1}$  such that Conditions (a)–(f) from Case 2 are satisfied. Define  $B_{m+1} = B$  and  $\beta_m = \beta^*$ . Obviously, the sequences  $A_1, \dots, A_n$  and  $B_1, \dots, B_{m+1}$ , the role  $R$  and the sequence  $\beta_1, \dots, \beta_m$  of regular roles witness that Case 2 is true for  $A \equiv \exists\beta^*.B$  and  $d, e$ .  $\square$

**Lemma 25** *Given an  $\mathcal{ALCOIF}_{reg}^2$ -TBox  $\mathcal{T}$  and a set  $t$  of concept names that occur in  $\mathcal{T}$ , it can be decided in EXP-TIME whether  $t$  is a type for  $\mathcal{T}$ .*

**Proof.** It can easily be checked in polynomial time that  $t$  satisfies all concept definitions in  $\mathcal{T}_{bool}$ . For checking Condition (P) note that in the proof of Lemma 24, we established a bound on the length on the numbers  $n, m$  that occur in (P). More precisely, we have  $n + m \leq \|\alpha\|$ . Given  $t$ ,

we can thus just enumerate all (exponentially many) potential witnesses for Conditions 1 and 2 of (P). It remains to note that, given a candidate  $A_1, \dots, A_n$  (for Condition 1) or  $A_1, \dots, A_n, B_0, \dots, B_m, R, \beta_1, \dots, \beta_{m-1}$  (for Condition 2), it can be checked in polynomial time whether it actually witnesses (P).  $\square$

**Lemma 20** *One can construct in time single exponential in  $\|\mathcal{T}\|$  a 2ABTA $_{\exists PP}$   $\mathfrak{A}_{\mathcal{T}}$  such that  $L(\mathfrak{A}_{\mathcal{T}})$  consists of exactly those binary well-formed  $\Sigma$ -labeled trees  $(T, \tau)$  with  $\mathcal{I}_{\mathcal{T}} \models \mathcal{T}$ .*

**Proof.** As it was argued in the main part,  $\mathfrak{A}_1$  and  $\mathfrak{A}_2$  are rather straightforward to construct; we omit the details.

The automaton  $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$  is similar to the automata  $\mathfrak{A}_r$ ,  $r \in \text{Fn}(\mathcal{T})$  from the proof of Lemma 17. In fact,  $Q$  also contains states  $q_0, q^*$  and states of the form

$$q_{\mathfrak{B}, B}, q_{\mathfrak{B}, B}^a, q_{\mathfrak{B}, B}^{a, X}, \bar{q}_{\mathfrak{B}, B}, \bar{q}_{\mathfrak{B}, B}^a, \bar{q}_{\mathfrak{B}, B}^{a, X}$$

where  $\mathfrak{B}$  is a finite automaton from a set  $\mathbf{B}$  (to be defined),  $s$  is a state in  $\mathfrak{B}$ ,  $a$  is an alphabet symbol in  $\mathfrak{B}$ , and  $X \in \{\uparrow, \downarrow, \text{src}, \text{tgt}\}$ . The set  $\mathbf{B}$  contains for every regular role  $\alpha = R\beta_1 \dots \beta_m$  such that  $R$  is a role that occurs in  $\mathcal{T}$ ,  $m \leq \|\mathcal{T}\|$ , and  $\beta_1, \dots, \beta_m$  are regular roles that occur in  $\mathcal{T}$ , the finite automaton  $\mathfrak{B}_\alpha$  equivalent to  $\alpha$  that exists due to Lemma 22. Additionally,  $Q$  contains states of the form

$$q_{t, \mathfrak{B}, B}^R, \bar{q}_{t, \mathfrak{B}, B}^R, q_{A \equiv \exists \alpha, B}, \bar{q}_{A \equiv \exists \alpha, B}$$

where  $t$  is a type for  $\mathcal{T}$ ,  $R$  is a role that occurs in  $\mathcal{T}$ , and  $\exists \alpha, B$  is a concept that occurs in  $\mathcal{T}$ . The Büchi condition  $\Omega$  consists of all states of the form  $\bar{q}_{\mathfrak{B}, B}^*$ .

The transition function  $\delta$  is also very similar to the transition functions  $\delta_r$  from the automata  $\mathfrak{A}_r$  in Lemma 17. In fact, the transitions for  $q_0$  remain the same, that is,

$$\begin{aligned} \delta(q_0, \circ) &= \square q_0 \\ \delta(q_0, \sigma) &= q^* \wedge \square q_0 \quad \text{for all } \sigma \neq \circ \end{aligned}$$

The transitions for  $q^*$  slightly change since  $\mathfrak{A}$  treats all roles in the same automaton. Thus, we include the following transitions for all  $(R, t, L) \in \Sigma$ :

$$\begin{aligned} \delta(q^*, (R, t, L)) &= \bigwedge_{A \in t, A \equiv \exists \alpha, B \in \mathcal{T}} q_{A \equiv \exists \alpha, B} \wedge \\ &\quad \bigwedge_{A \notin t, A \equiv \exists \alpha, B \in \mathcal{T}} \bar{q}_{A \equiv \exists \alpha, B} \end{aligned}$$

If the automaton visits a node  $n$  in state  $q_{A \equiv \exists \alpha, B}$ , this represents the obligation to verify that in the represented interpretation the domain element encoded by  $n$  satisfies  $\exists \alpha, B$ . (Again, states  $\bar{q}_{A \equiv \exists \alpha, B}$  behave complementary, so we concentrate on  $q_{A \equiv \exists \alpha, B}$ .) The difficulty here is that one domain element in the represented interpretation is encoded in different ‘slices’ of the input tree. We have thus to be careful in which of the slices the automaton verifies that  $\exists \alpha, B$  is satisfied. This is where we use the additional property (P). Intuitively, the role  $R$  in Condition 2 of (P) determines the first (non-test) step and thus the slice.

Let  $(S, t, L) \in \Sigma$  with  $S \in \{r, r^-\}$  for some role name  $r$  and  $A \equiv \exists \alpha, B \in \mathcal{T}$  with  $A \in t$ . Since  $t$  is a type for

$\mathcal{T}$ , Condition 1 or Condition 2 of (P) is satisfied for  $A$ ,  $A \equiv \exists \alpha, B$ . We define  $\delta(q_{A \equiv \exists \alpha, B}, (S, t, L))$  by a case distinction as follows:

- If Condition 1 of (P) is satisfied for  $A$ ,  $A \equiv \exists \alpha, B$ , then set

$$\delta(q_{A \equiv \exists \alpha, B}, (S, t, L)) = \text{true}.$$

- Otherwise, let us define a set  $\mathbf{B}' \subseteq \mathbf{B}$  as follows. If  $\mathfrak{B} \in \mathbf{B}$  is equivalent to the regular role  $R\beta_1 \dots \beta_{m-1}$ , then  $\mathfrak{B} \in \mathbf{B}'$  iff there are concept names  $A_1, \dots, A_n$  and  $B_0, \dots, B_m$  such that  $m + n \leq \|\alpha\|$ ,

$$R \in \{r, r^-\} \cup \{s, s^- \mid s \text{ occurs in } \mathcal{T} \text{ and } s \notin \text{Fn}(\mathcal{T})\},$$

and Conditions 2(a)–(d) of (P) are satisfied. Then, we define:

$$\delta(q_{A \equiv \exists \alpha, B}, (S, t, L)) = \begin{cases} \text{true} & \text{if } \mathbf{B}' = \emptyset, \\ \bigvee_{\mathfrak{B} \in \mathbf{B}'} q_{\mathfrak{B}, B} & \text{otherwise.} \end{cases}$$

The fact that the empty disjunction in the second case corresponds to true can be explained as follows: if  $\mathbf{B}'$  is empty, then the concept  $\exists \alpha, B$  has to be verified in a different slice. Condition (P) on types ensures that such a slice exists.

Just like the automata  $\mathfrak{A}_r$  from Lemma 17, visiting a node  $n$  in state  $q_{\mathfrak{B}, B}$  represents the obligation to find a node  $n'$  which satisfies  $B$  and is connected to  $n$  via a  $\mathfrak{B}$ -path. Recall that it does so by non-deterministically guessing the steps of this path according to the finite automaton  $\mathfrak{B}$ . In fact, the transition function for states of the form  $q_{\mathfrak{B}, B}$  (and the states reachable from that) is identical to the transition function  $\delta_r$  in the automaton  $\mathfrak{A}_r$ , except for inputs of the form  $(q_{\mathfrak{B}, B}^R, (S, t, L))$  when  $R \in \{r, r^-\}$  and  $S \in \{s, s^-\}$  for role names  $r \neq s$ . Intuitively, these inputs represent the situation that the automaton visits a node in the  $s$ -slice and tries to make an  $R$ -step,  $R \in \{r, r^-\}$  and  $r \neq s$ . This situation cannot occur in a successful run of  $\mathfrak{A}_r$  on a well-formed input tree, and hence the definition of  $\delta_r(q, \sigma)$  for such  $q, \sigma$  is actually irrelevant. Here This is, however, not the case for the stronger fragment.

Before we give the transition function in this case, let us note that the above definition of  $\delta(q_{A \equiv \exists \alpha, B}, (S, t, L))$  makes sure that the automaton never attempts to make an  $S$ -step,  $S \in \{s, s^-\}$ ,  $s \in \text{Fn}(\mathcal{T})$  when it visits a node in the slice for a functional role  $r \neq s$ . Indeed, this is the case for the first step since the first step of every automaton  $\mathfrak{B} \in \mathbf{B}'$  will either be  $r, r^-,$  or  $s$  or  $s^-$ , for some non-functional role  $s$ . To see this for later steps, let  $R\beta_1 \dots \beta_{m-1}$  be the regular role for which  $\mathfrak{B}$  was added to  $\mathbf{B}'$ , in processing the concept  $\exists \alpha, B$ . By Condition 2(a) of (P), there are concept names  $A_1, \dots, A_n$  such that  $L(A_1 \dots A_n R\beta_1 \dots \beta_{m-1}) \subseteq L(\alpha)$ . Since  $\alpha$  satisfies  $(*_2)$ , so does  $R\beta_1 \dots \beta_{m-1}$ . Thus, the automaton  $\mathfrak{B}$  does not allow consecutive steps for distinct functional roles.

For the definition of  $\delta$  on the mentioned inputs, let  $R \in \{r, r^-\}$ . Then for every state  $q_{\mathfrak{B}, B}^R$  and every symbol  $(S, t, L) \in \Sigma$  with  $S \in \{s, s^-\}$  for some  $s \neq r$  such that  $s \notin \text{Fn}(\mathcal{T})$ , we set

$$\delta(q_{\mathfrak{B}, B}^R, (S, t, L)) = \bigvee_{t' \in \mathbf{T}} q_{t', \mathfrak{B}, B}^R$$

where  $\mathbf{T}$  is the set of all types  $t'$  for  $\mathcal{T}$  such that:

1. for all CIs  $A \equiv \exists R.B \in \mathcal{T}$ ,  $B \in t'$  implies  $A \in t$ ;
2. for all CIs  $A \equiv \exists R^-.B \in \mathcal{T}$ ,  $B \in t$  implies  $A \in t'$ .

That is, in the represented interpretation, nodes with types  $t$  and  $t'$  will be connected by  $R$ . In a state  $q_{t', \mathfrak{B}, B}^R$ , the automaton non-deterministically navigates to a node with label  $(S, t', L)$  and changes to state  $q_{\mathfrak{B}, B}$ ; we omit the details.

Using the arguments provided during the definition of the transition function  $\delta$  above, it is routine to verify that  $\mathfrak{A}$  accepts a well-formed tree  $\tau$  iff the represented interpretation  $\mathcal{I}_\tau$  is a model of  $\mathcal{T}$ . It can also be verified that the construction can be carried out in exponential time.  $\square$

## Proof of Theorem 21

**Theorem 21** (Finite) *satisfiability in  $\mathcal{ALCCOLIF}_{reg}$  can be reduced in polynomial time to (finite) satisfiability in  $\mathcal{ALCCOLIF}_{reg, qual}^1$ .*

**Proof.** Let  $\mathcal{T}$  be an  $\mathcal{ALCCOLIF}_{reg}$ -TBox. Suppose without loss of generality that some nominal occurs in  $\mathcal{T}$ . The general idea is to replace role names  $r$  in  $\mathcal{T}$  by the regular role  $s \cdot X_r \cdot ? \cdot s$ , where  $s$  and  $X_r$  are fresh names. That is, an  $r$ -step is simulated by two  $s$ -steps via an auxiliary element which satisfies a concept  $X_r$ . To make this idea work, we also need to relativize the TBox to the ‘real’ domain elements. Formally, we use the following fresh symbols:

- role name  $s$ ;
- concept names  $X_r$ , for each role name  $r$  that occurs in  $\mathcal{T}$ ;
- concept name  $D$ .

We denote with  $\delta(C)$  the concept that is obtained from  $C$  by replacing every role name  $r$  by the regular role  $s \cdot X_r \cdot ? \cdot s$ . Moreover, the *relativization*  $C|_D$  of a concept  $C$  to the concept name  $D$  is defined inductively as follows:

$$\begin{aligned} A|_D &= A \sqcap D \\ (\neg C)|_D &= D \sqcap \neg(C|_D) \\ (C_1 \sqcap C_2)|_D &= D \sqcap C_1|_D \sqcap C_2|_D \\ (\exists \alpha.C)|_D &= D \sqcap \exists \alpha|_D.C|_D \end{aligned}$$

where  $\alpha|_D$  is obtained from  $\alpha$  by replacing every test  $C?$  that occurs in  $\alpha$  by the test  $C|_D?$ .

We now define a TBox  $\mathcal{T}'$  by taking

$$\begin{aligned} \mathcal{T}' &= \{\delta(C_1)|_D \sqsubseteq \delta(C_2)|_D \mid C_1 \sqsubseteq C_2 \in \mathcal{T}\} \cup \\ &\quad \{\top \sqsubseteq (\leq 1 s X_r) \mid \text{func}(r) \in \mathcal{T}\} \cup \\ &\quad \{\top \sqsubseteq (\leq 1 s^- X_r) \mid \text{func}(r^-) \in \mathcal{T}\} \cup \\ &\quad \{\{a\} \sqsubseteq D \mid \{a\} \text{ occurs in } \mathcal{T}\} \cup \\ &\quad \{D \sqsubseteq \forall s. \neg D, \neg D \sqsubseteq \forall s. D\} \cup \\ &\quad \{\neg D \sqsubseteq (\leq 1 s \top) \sqcap (\leq 1 s^- \top)\} \end{aligned}$$

Intuitively, the first line captures the relativized TBox; the second and third lines capture the functionality assertions; the fourth line ensures that all nominals are contained in the intended domain  $D$ . In particular, due to the assumption that  $\mathcal{T}$  contains at least one nominal, the intended domain is not empty. Finally, the last two lines axiomatize the fact that  $D$

is the intended domain. Note that the only role name that occurs in  $\mathcal{T}'$  is  $s$ , so indeed  $\mathcal{T}'$  is an  $\mathcal{ALCCOLIF}_{reg, qual}^1$ -TBox.

*Claim.*  $\mathcal{T}$  is (finitely) satisfiable iff  $\mathcal{T}'$  is (finitely) satisfiable.

*Proof of the Claim.* ( $\Rightarrow$ ) Let  $\mathcal{I}$  be a model of  $\mathcal{T}$ . We define an interpretation  $\mathcal{J}$  as follows:

$$\begin{aligned} \Delta^{\mathcal{J}} &= \Delta^{\mathcal{I}} \cup \{r(d, e) \mid r \in \mathbf{N}_R, (d, e) \in r^{\mathcal{I}}\} \\ A^{\mathcal{J}} &= A^{\mathcal{I}} \quad \text{for all } A \text{ that occur in } \mathcal{T} \\ D^{\mathcal{J}} &= \Delta^{\mathcal{I}} \\ X_r^{\mathcal{J}} &= \{r(d, e) \mid (d, e) \in r^{\mathcal{I}}\} \quad \text{for all } r \text{ that occur in } \mathcal{T} \\ s^{\mathcal{J}} &= \{(d, r(d, e)), (r(d, e), e) \mid r(d, e) \in \Delta^{\mathcal{J}}\} \end{aligned}$$

It is routine to verify that  $\mathcal{J}$  is a model of  $\mathcal{T}'$ , and  $\mathcal{J}$  is finite iff  $\mathcal{I}$  is finite.

( $\Leftarrow$ ) Conversely, let  $\mathcal{I}$  be a model of  $\mathcal{T}'$ . We define an interpretation  $\mathcal{J}$  by taking

$$\begin{aligned} \Delta^{\mathcal{J}} &= D^{\mathcal{I}} \\ A^{\mathcal{J}} &= A^{\mathcal{I}} \\ r^{\mathcal{J}} &= \{(d, e) \in \Delta^{\mathcal{J}} \times \Delta^{\mathcal{J}} \mid \text{there exists an } f \in X_r^{\mathcal{I}} \\ &\quad \text{such that } (d, f), (f, e) \in s^{\mathcal{I}}\} \end{aligned}$$

for all concept names  $A$  and all role names  $r$  that occur in  $\mathcal{T}$ . It can be verified that  $\mathcal{J}$  is a model of  $\mathcal{T}$ , and  $\mathcal{J}$  is finite iff  $\mathcal{I}$  is finite.  $\square$