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

Jean Christoph Jung[1] , **Carsten Lutz**[1] , **Thomas Zeume**[2]
[1]University of Bremen, Germany
[2]Ruhr University Bochum, Germany
{jeanjung,clu}@uni-bremen.de, thomas.zeume@rub.de

Abstract

We consider fragments of the description logic $\mathcal{SHOIF}$ extended with regular expressions on roles. Our main result is that satisfiability and finite satisfiability are decidable in two fragments $\mathcal{SHOIF}^1_{reg}$ and $\mathcal{SHOIF}^2_{reg}$, 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 α^*. $\mathcal{SHOIF}^1_{reg}$ encompasses the extension of $\mathcal{SHOIF}$ 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 accessed July 2 2020). In its essence, OWL 2 DL is identical to a very expressive DL called $\mathcal{SROIQ}$ (Horrocks, Kutz, and Sattler 2006). An important feature of $\mathcal{SROIQ}$ and related DLs such as $\mathcal{SHOIF}$ and $\mathcal{ALCOIF}$ 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 $\mathcal{ALCOIF}$ 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 $\mathcal{ALCOIF}$ with fixed points, a notational variant of the μ-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

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

where '$+$' denotes role union and '$\cdot^+$' transitive closure of a role. The extension $\mathcal{ALCOIF}_{reg}$ of the aforementioned $\mathcal{ALCOIF}$ 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 μ-calculus discussed above. Another one is the extension with ω-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 $\mathcal{SHOIQ}$ with transitive closure, but there are problems in the construction and a corrected version has not yet been published (Le Duc, C., personal communication, January 2019).

We study fragments of the description logic $\mathcal{SHOIF}_{reg}$, that is $\mathcal{SHOIF}$ 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 $\mathcal{SHOIF}^1_{reg}$, every regular (sub)role expression of the form α^* contains either no functional role or only a single role name (and possibly its inverse). In the more expressive $\mathcal{SHOIF}^2_{reg}$, every regular (sub)role expression α^* 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 β a composition of tests. In addition, both fragments disallow subroles of functional roles. We show that satisfiability in $\mathcal{SHOIF}^1_{reg}$ is decidable and NEXPTIME-complete and thus not harder than in $\mathcal{SHOIF}$ without regular expressions, and that satisfiability in $\mathcal{SHOIF}^2_{reg}$ is decidable in 2NEXPTIME. We establish the same results for finite satisfiability which does not coincide with the unrestricted case. Note that $\mathcal{SHOIF}^1_{reg}$ contains the extension $\mathcal{SHOIF}^+$ of $\mathcal{SHOIF}$ with transitive closure of roles and thus our results imply that satisfiability and finite satisfiability in $\mathcal{SHOIF}^+$ 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 $\mathcal{SHOIF}^1_{reg}$ and $\mathcal{SHOIF}^2_{reg}$ 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^2, the guarded two-variable fragment of first-order logic with counting quantifiers (Pratt-Hartmann 2007). GC^2 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 $\mathcal{ALCOIF}$, a very significant fragment of GC^2. In fact, we can express in $\mathcal{ALCOIF}_{reg}$ that a role is an equivalence relation or a linear order, using fresh helper symbols in the latter case. Consequently, $\mathcal{ALCOIF}$ extended with these semantic properties (even simultaneously) is still decidable and NEXPTIME-complete. While decidability results for GC^2 and its non-guarded extension C^2 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 $\mathcal{ALCOIQ}$ extended with forests is decidable. Our results capture the $\mathcal{ALCOIF}$ 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 $\mathsf{N_C}$, $\mathsf{N_R}$, and $\mathsf{N_I}$ be countably infinite sets of *concept, role, and individual names*. $\mathcal{ALCOIF}_{reg}$*-concepts* C, *roles* R, and *regular roles* α 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 $\mathsf{N_C}$, a over $\mathsf{N_I}$, and r over role names from $\mathsf{N_R}$. A role of the form r^- is called an *inverse role*. We write $\bot$ as an abbreviation for $A \sqcap \neg A$ with A a fixed concept name, $\top$ for $\neg\bot$, $C \sqcup D$ 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 $\mathcal{SHOIF}_{reg}$-TBox $\mathcal{T}$ is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$ with C, D $\mathcal{ALCOIF}_{reg}$-concepts, *role inclusions (RIs)* $R \sqsubseteq S$ with R, S roles, and *functionality assertions* $\mathsf{func}(R)$ with R a role. We use $\mathsf{Fn}(\mathcal{T})$ to denote the set of role names r such that $\mathcal{T}$ contains $\mathsf{func}(r)$ or $\mathsf{func}(r^-)$. $\mathcal{SHOIF}$-TBoxes typically also admit transitivity assertions $\mathsf{trans}(r)$, r a role name. This is syntactic sugar in $\mathcal{SHOIF}_{reg}$ 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 $\mathcal{ALCOIF}_{reg}$*-TBox* is a $\mathcal{SHOIF}_{reg}$-TBox that does not contain role inclusions. The letter $\mathcal{F}$ and subscript $\cdot_{reg}$ in $\mathcal{ALCOIF}_{reg}$ and in $\mathcal{SHOIF}_{reg}$ indicate the presence of functionality assertions and of regular roles. It should thus be clear what we mean, for example, with $\mathcal{ALCOI}_{reg}$*-concepts* and with $\mathcal{ALCOIF}$*-TBoxes*. We use $\mathcal{SHOIF}^+$ to denote the extension of $\mathcal{SHOIF}$ with transitive closure of roles, that is, the fragment of $\mathcal{SHOIF}_{reg}$ 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 $\mathsf{func}(R)$ 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 $\mathcal{ALCOIF}$ does not have the finite model property, that is, finite and unrestricted satisfiability do not coincide. For example, the $\mathcal{ALCOIF}$-TBox

$$\top \sqsubseteq \exists s.A, \;\; A \sqsubseteq \forall r^-.\bot, \;\; \top \sqsubseteq \exists r.\top, \;\; \mathsf{func}(r^-)$$

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.

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

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

$(*_1)$ *if a role name $r \in \mathsf{Fn}(\mathcal{T})$ occurs in α outside of tests, then no other role name occurs in α outside of tests.*

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

$(*_2)$ *no $w \in L(\alpha^*)$ contains an infix of the form $R\beta S$ with $R \in \{r, r^-\}$, $S \in \{s, s^-\}$, $r, s \in \mathsf{Fn}(\mathcal{T})$, $r \neq s$, and β a composition of tests.*

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

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

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

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

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 $r \in \mathsf{Fn}(\mathcal{T})$ 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 $r \in \mathsf{Fn}(\mathcal{T})$.

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

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

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

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

(5) The TBox

$$\begin{array}{ll} \top \sqsubseteq \exists r^*.\{a\} \sqcup \exists(r^-)^*.\{a\}, & \{a\} \sqsubseteq \neg\exists r^+.\{a\}, \\ \mathsf{func}(r), & \mathsf{func}(r^-) \end{array}$$

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 $\mathcal{SHOIF}^i_{reg}$ can be reduced in polynomial time to (finite) satisfiability in $\mathcal{ALCOIF}^i_{reg}$.*

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 *$\mathcal{ALCOIF}^i_{reg}$-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:
$$\begin{array}{lll} A \equiv \{a\} & A \equiv \neg B & A \equiv B \sqcup B' \\ A \equiv B \sqcap B' & A \equiv \exists\alpha.B & \end{array}$$
2. if $\mathsf{func}(r^-)$ occurs in $\mathcal{T}$, then so does $\mathsf{func}(r)$, for every role name r;
3. if a regular role α occurs in $\mathcal{T}$, then α satisfies Condition $(*_i)$ 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 $(*_i)$ needs to be satisfied for every regular role α that occurs in $\mathcal{T}$ rather than only for those α that occur in the form α^*. Conditions (5)-(8) are inspired by the Fischer-Ladner closure in PDL (Fischer and Ladner 1979). Every $\mathcal{ALCOIF}^i_{reg}$-TBox in normal form can be viewed as an $\mathcal{ALCOIF}^i_{reg}$-TBox according to the original definition by reading concept definitions $A \equiv C$ as an abbreviation for the CI $\top \sqsubseteq (\neg A \sqcup C) \sqcap (\neg C \sqcup A)$. The following lemma shows that we can work with TBoxes in normal form without loss of generality.

Proposition 6 *Every $\mathcal{ALCOIF}^i_{reg}$-TBox $\mathcal{T}$ can be converted in polynomial time into an $\mathcal{ALCOIF}^i_{reg}$-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 Σ be a finite alphabet. A *Σ-labeled tree* is a pair (T, τ) where T is a tree and $\tau : T \to \Sigma$. We may write only τ for the Σ-labeled tree (T, τ) 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 ψ 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* φ if $(\mathbb{N} \cup \{\aleph_0\}, 0, 1, \aleph_0, +) \models \varphi(\mathbf{a})$ where $|\mathbf{x}|$ denotes the length of $\mathbf{x}$. Further, φ 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 (2ABTA$_{\exists PP}$) 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, δ *a* transition function, $\Omega \subseteq Q$ *a set of* recurring states, *and φ_c an existential Presburger formula with free variables $x_{\sigma_1}, \dots, x_{\sigma_m}$. The transition function δ 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 $p, \Diamond p, \Box p, \Diamond^- p, \Box^- p$ with $p \in Q$.*

A run *of a 2ABTA$_{\exists PP}$ $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$ on a k-ary Σ-labeled tree (T, τ) 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 θ such that S satisfies θ 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 $\Box p \in S$ 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 $\Box^- p \in S$, 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 $\gamma = i_0 i_1 \cdots$ 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, τ) if*

1. *there exists an accepting run of $\mathcal{A}$ on (T, τ);*
2. *$\#_{\sigma_1}(\tau), \dots, \#_{\sigma_m}(\tau)$ satisfies φ_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 Σ-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 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 2ABTA$_{\exists PP}$: 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 2ABTA$_{\exists PP}$ $\mathfrak{A}_i = (Q_i, \Sigma, q_{0,i}, \delta_i, \Omega_i, \varphi_{c,i})$, $i \in \{1, 2\}$, one can construct in time polynomial in $||\mathcal{A}_1|| + ||\mathcal{A}_2||$ a 2ABTA $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$ 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 2ABTA$_{\exists PP}$ is to decide, given a 2ABTA$_{\exists PP}$ $\mathfrak{A}$, whether $L(\mathfrak{A}) \neq \emptyset$ (resp. $L_f(\mathfrak{A}) \neq \emptyset$).

Theorem 9 *Non-emptiness of 2ABTA$_{\exists PP}$ is decidable in* NExpTime. *More precisely, there is a non-deterministic algorithm that, given a 2ABTA$_{\exists PP}$ $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$, 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 2ABTA$_{\exists PP}$ as future work.

5 Non-Emptiness of 2ABTA$_{\exists PP}$: 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$_{\exists PP}$ 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, τ) 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$_{\exists PP}$ $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$ 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

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

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 σ, then it can send the states $q_1, \dots, q_k$ to the k successors of the current node. A transition (q, σ) 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 Σ as the terminal symbols, and each transition $(q, \sigma, q_1, \dots, q_m)$ as a grammar rule $q \to \sigma q_1 \cdots q_m$. 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 φ_G over $\mathbb{N}$ iff there is a word $w \in L(G)$ such that $\mathbf{a} = \#_{\sigma_1}(w), \dots, \#_{\sigma_n}(w)$.*[1]

[1]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$_{\exists PP}$: 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 $\Sigma_{\mathit{fin}} = \{\sigma_1, \dots, \sigma_n\} \subseteq \Sigma$. There is a finite set $\mathcal{P}_{\mathfrak{A}, \Sigma_{\mathit{fin}}}$ 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 $\varphi \in \mathcal{P}_{\mathfrak{A}, \Sigma_{\mathit{fin}}}$ 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 $\sigma \in \Sigma \setminus \Sigma_{\mathit{fin}}$.

Moreover, there is a non-deterministic procedure that given $\mathfrak{A}$ and Σ_{fin} generates exactly the formulas in $\mathcal{P}_{\mathfrak{A}, \Sigma_{\mathit{fin}}}$ 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$_{\exists PP}$ $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega, \varphi_c)$ 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 $\Sigma_{\mathit{fin}} = \{\sigma_{i_1}, \dots, \sigma_{i_n}\} \subseteq \Sigma$ and use the non-deterministic procedure from Lemma 12 to guess a sentence $\psi \in \mathcal{P}_{\mathfrak{A}', \Sigma_{\mathit{fin}}}$ 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

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

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, Σ, q_0, Δ) 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 $\Sigma_{\mathit{fin}} \subseteq \Sigma$. 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, τ) 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 Σ_{fin} and (2) all leaf nodes are labeled with symbols from $\Sigma_{\mathit{inf}} := \Sigma \setminus \Sigma_{\mathit{fin}}$. 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 Σ_{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* Σ_{inf} is a triple $(q, \sigma, \Psi) \in Q \times \Sigma_{inf} \times 2^{\Sigma_{inf}}$. We say that (q, σ, Ψ) is *satisfiable* if there is a Σ_{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, τ) 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, σ, Ψ) *for* $\mathfrak{A}$ *and* Σ_{inf}*, it is decidable in* NP *whether* (q, σ, Ψ) *is satisfiable.*

The proof of Lemma 13 uses a non-emptiness check for NBTAs. Inn fact, we show that there is a finite set $\mathbf{B}$ of NBTAs such that (q, σ, Ψ) 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* Σ_{inf} is a set

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

such that each (q_i, σ_i, Ψ_i) is an obligation for $\mathfrak{A}'$ and Σ_{inf} with $\Psi_1, \ldots, \Psi_k$ a partition of Σ_{inf}, some Ψ_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 φ_S for each satisfiable obligation pattern $S = \{(q_1, \sigma_1, \Psi_1), \ldots, (q_k, \sigma_k, \Psi_k)\}$ for $\mathfrak{A}'$ and Σ_{inf}. To define φ_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 Σ_S-labeled trees (T, τ) for which there is a Σ-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 τ' such that $\tau(n) = (r(n), \tau'(n))$ for every leaf n of T.

Let $\Sigma_{fin} = \{\sigma_1, \ldots, \sigma_n\}$ and for uniformity, $\Sigma_S = \{\sigma_1, \ldots, \sigma_{n+m}\}$. We can view $\mathfrak{A}_S$ as a CFG G and obtain a Presburger formula φ_G from Theorem 11 that has free variables $x_{\sigma_1}, \ldots, x_{\sigma_{n+m}}$. We construct from φ_G the desired sentence φ_S with free variables $x_{\sigma_1}, \ldots, x_{\sigma_n}$. 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, σ, Ψ). Then

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

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 Σ_{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 δ 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 Σ_{inf}. It next uses Lemma 13 to verify that S is satisfiable and then generates and outputs the formula φ_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* $\mathcal{ALCOIF}^1_{reg}$*, satisfiability and finite satisfiability are* NExpTime*-complete.*

The lower bounds stated in Theorem 14 already apply to $\mathcal{ALCOIF}$ 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 $\mathcal{ALCOIF}^1_{reg}$-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 $r \in \mathsf{Fn}(\mathcal{T})$ occurs in α, and
- $\mathcal{T}_r$ to denote the set of concept definitions $A \equiv \exists \alpha.B$ in $\mathcal{T}$ such that $r \in \mathsf{Fn}(\mathcal{T})$ is the only role name that occurs in α, plus $\mathcal{T} \cap \{\mathsf{func}(r), \mathsf{func}(r^-)\}$.

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

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 $\mathsf{tp}_{\mathcal{I}}(d)$. Let $\mathsf{tp}(\mathcal{T})$ 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 $\{\mathsf{tp}_{\mathcal{I}}(d) \mid d \in \Delta^{\mathcal{I}}\}$ and for each $t \in \mathsf{tp}(\mathcal{T})$, we denote with $\#_t(\mathcal{I})$ the cardinality of the set $\{d \in \Delta^{\mathcal{I}} \mid \mathsf{tp}_{\mathcal{I}}(d) = t\}$. We now give the characterization.

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

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

(a) $\mathcal{I}_r$ *realizes* Γ*;*

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

Notice that the models $\mathcal{I}_r$, $r \in \mathsf{Fn}(\mathcal{T})$, 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* $\mathcal{ALCOIF}^1_{reg}$*-TBox* $\mathcal{T}$ *in normal form and a set* $\Gamma \subseteq \mathsf{tp}(\mathcal{T})$*, we 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*

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 $\Gamma \subseteq \mathsf{tp}(\mathcal{T})$ and verify Conditions 1 and 2 from Proposition 15. Condition 1 is equivalent to satisfiability of the $\mathcal{ALCOI}_{reg}$-TBox

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

where $\widehat{r}$ is a fresh role name and $\bigsqcap t$ 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 $\mathcal{ALCOI}_{reg}$ 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 $\mathcal{ALCOIF}^1_{reg}$-TBox in normal form and let $\Gamma \subseteq \mathsf{tp}(\mathcal{T})$. Take any (finite or infinite) model $\mathcal{I}_r$ of $\mathcal{T}_{bool} \cup \mathcal{T}_r$, $r \in \mathsf{Fn}(\mathcal{T})$. 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 $\mathsf{func}(r) \in \mathcal{T}$ and $\mathsf{func}(r^-) \in \mathcal{T}$, 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 $\mathsf{func}(r^-) \in \mathcal{T}$ and $\mathsf{func}(r) \notin \mathcal{T}$. 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 2ABTA$_{\exists PP}$ 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 $\mathcal{I}_r$, $r \in \mathsf{Fn}(\mathcal{T})$, as a tree that can be the input to a 2ABTA$_{\exists PP}$. Since there

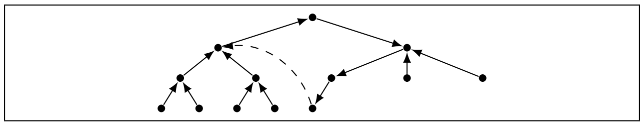
Figure 1: Example component of a graph $(\Delta^\mathcal{I}, r^\mathcal{I})$, $\mathsf{func}(r) \in \mathcal{T}$. Removing the dashed edge turns it into a tree.

is no bound on the outdegree of these models, we use binary trees and a suitable encoding. Let $\mathcal{R}$ denote the set $\{r, r^- \mid r \in \mathsf{Fn}(\mathcal{T})\}$ and $\mathcal{L} = 2^{\{\mathsf{src},\mathsf{tgt}\}}$. 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 Σ-labeled tree (T, τ) is *well-formed* if it satisfies the following conditions:

1. for each $r \in \mathsf{Fn}(\mathcal{T})$, 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 $r \in \mathsf{fn}(\mathcal{T})$, 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 $r \in \mathsf{Fn}(\mathcal{T})$ 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 $\mathsf{src} \in L_1$ and a unique descendant n_2 of n with $\tau(n_2) = (R_2, t_2, L_2)$ and $\mathsf{tgt} \in L_2$, or there is no descendant of either kind.

Condition 1 ensures that the interpretations $\mathcal{I}_r$ have nonempty domains. Condition 2 separates the representations of the interpretations $\mathcal{I}_r$, $r \in \mathsf{Fn}(\mathcal{T})$: 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 $r \in \mathsf{Fn}(\mathcal{T})$.

Every well-formed Σ-labeled tree (T, τ) gives rise to a collection of interpretations $\mathcal{I}^\tau_r$, $r \in \mathsf{Fn}(\mathcal{T})$, as follows:

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

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

$$s^{\mathcal{I}_r^\tau}=\emptyset \quad \text{if } s\neq r,$$

for all $a\in\mathsf{N_I}$, $A\in\mathsf{N_C}$, and $s\in\mathsf{N_R}$. Note that it is not (yet) guaranteed that $\mathcal{I}_r^\tau$ 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 $r\in\mathsf{Fn}(\mathcal{T})$, such that Conditions 2a and 2b of Proposition 15 are satisfied. Then there is a binary well-formed Σ-labeled tree (T,τ) with $\mathcal{I}_r^\tau=\mathcal{I}_r$ for all $r\in\mathsf{Fn}(\mathcal{T})$. We construct such a (T,τ) 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 ε. We put $\tau(\varepsilon)=\circ$. As already mentioned, each graph $(\Delta^{\mathcal{I}_r},r^{\mathcal{I}_r})$, $r\in\mathsf{Fn}(\mathcal{T})$, 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 $(\Delta^{\mathcal{I}_r},r^{\mathcal{I}_r})$, $r\in\mathsf{Fn}(\mathcal{T})$. Assume that all these components are linearly ordered. For $1\leq\ell\leq m$, we represent the ℓ-th such component $(\Delta^{\mathcal{J}},r^{\mathcal{J}})$ using the subtree of (T,τ) rooted at the successor ℓ 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 π between T_ℓ 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 ℓn, $n\in T_\ell$, and put $\tau(\ell ni)=(R,t,L)$ where $R=r$ if $(\pi(n),\pi(ni))\in r^{\mathcal{J}}$ and $R=r^-$ otherwise, $t=\mathsf{tp}_{\mathcal{J}}(\pi(ni))$, and L contains src iff $\pi(ni)=d_0$ and tgt iff $\pi(ni)=e_0$. We define $\tau(\ell)$ in the same way using ε 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 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 Σ-labeled trees (T,τ) such that for $\mathcal{I}_r=\mathcal{I}_r^\tau$, $r\in\mathsf{Fn}(\mathcal{T})$, Condition 2 of Propositon 15 is satisfied.*

Proof. (sketch) The 2ABTA$_{\exists PP}$ $\mathfrak{A}_\mathcal{T}$ is the intersection of several 2ABTA$_{\exists PP}$, namely $\mathfrak{A}_1$, $\mathfrak{A}_2$, $\mathfrak{A}_3$, and $\mathfrak{A}_r$ for each $r\in\mathsf{Fn}(\mathcal{T})$. While $\mathfrak{A}_1$ checks that (T,τ) is well-formed, $\mathfrak{A}_2$ verifies that the $\mathcal{I}_r^\tau$ satisfy Conditions 2a and 2b from Proposition 15 and $\mathfrak{A}_3$ makes sure that $R^{\mathcal{I}_r^\tau}$ is a partial function whenever $\mathsf{func}(R)\in\mathcal{T}$, $R\in\{r,r^-\}$. $\mathfrak{A}_1$ 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,τ) 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 $n^{\uparrow\uparrow}$ defined;
- a node $n\in T$ such that $\tau(n)=(r,t,L)$, $\mathsf{src}\in L$, and $n^\uparrow$ defined;
- nodes n_1,n_2 such that $\tau(n_i)=(r^-,t_i,L_i)$ for $i\in\{1,2\}$, $\mathsf{func}(r^-)\in T$, 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 $\mathcal{I}_r^\tau$ satisfies the CIs from $\mathcal{T}_r$. It sends a copy of itself to every node n of the input tree (T,τ) 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 $n\in(\exists\alpha.B)^{\mathcal{I}_r^\tau}$. The automaton thus needs to verify the (non-)existence of an α-path that starts at n and ends in a node whose type includes B. This is implemented by representing α as a finite automaton $\mathcal{B}$ on finite words, tracing runs of $\mathcal{B}$ through the $\mathcal{I}_r^\tau$ part of (T,τ), and making use of the Büchi acceptance condition. ❏

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 $\mathcal{SHOIF}^1_{reg}$-TBox $\mathcal{T}$ has a model of size at most double exponential in $||\mathcal{T}||$. This bound is tight since there are $\mathcal{ALCOIF}$-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 $\mathcal{ALCOIF}^2_{reg}$, satisfiability and finite satisfiability are decidable in* 2NExpTime.

A 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 $\mathcal{T}_{bool}$, $\mathcal{T}_{reg}$, $\mathcal{T}_r$, $r\in\mathsf{Fn}(\mathcal{T})$, since regular expressions that contain a role name $r\in\mathsf{Fn}(\mathcal{T})$ 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 2ABTA$_{\exists PP}$.

Proposition 19 *Given an $\mathcal{ALCOIF}^2_{reg}$-TBox $\mathcal{T}$ in normal form, we can construct in time single exponential in $||\mathcal{T}||$ a 2ABTA$_{\exists PP}$ $\mathfrak{A}_\mathcal{T}$ 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 $\mathcal{ALCOIF}^2_{reg}$-TBox in normal form whose satisfiability is to be decided. We use Σ-labeled trees for the alphabet $\Sigma=\{\circ\}\cup(\mathcal{R}\times\mathsf{tp}(\mathcal{T})\times\mathcal{L})$, $\mathcal{R}$ 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 $t\in\mathsf{tp}(\mathcal{T})$ and $r_1,r_2\in\mathsf{Fn}(\mathcal{T})$,

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

In the previous section, a Σ-labeled tree (T,τ) encodes a collection of interpretations $\mathcal{I}_r^\tau$, $r\in\mathsf{Fn}(\mathcal{T})$. Here, we want (T,τ) to encode a single interpretation $\mathcal{I}_\tau$ that is a model of $\mathcal{T}$. Essentially, what were distinct interpretations $\mathcal{I}_r^\tau$ previously are now 'slices' of $\mathcal{I}_\tau$. In particular, there is no one-to-one correspondence between the nodes in (T, τ) and the domain elements in $\mathcal{I}_\tau$ as every domain element of $\mathcal{I}_\tau$ is represented by multiple nodes in (T, τ), one for each $r \in \mathsf{Fn}(\mathcal{T})$ (each 'slice'). We next make this precise.

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

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

for all $a \in \mathsf{N_I}$, $A \in \mathsf{N_C}$, and $r \in \mathsf{Fn}(\mathcal{T})$. We still have to define $s^{\mathcal{I}_\tau}$ for all $s \in \mathsf{N_R} \setminus \mathsf{Fn}(\mathcal{T})$. 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 Σ-labeled tree (T, τ) with $\mathcal{I}_\tau = \mathcal{I}$. This is done as in the previous section, using $\mathcal{I}$ in place of each interpretation $\mathcal{I}_r$, $r \in \mathsf{Fn}(\mathcal{T})$—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 $s \in \mathsf{N_R} \setminus \mathsf{Fn}(\mathcal{T})$ are not represented explicitly in (T, τ), 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 Σ-labeled trees (T, τ) 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 $\mathsf{func}(R) \in \mathcal{T}$. The construction is similar to those of the automata $\mathfrak{A}_1$ and $\mathfrak{A}_3$ in the proof of Lemma 17. The automata $\mathfrak{A}_r$, $r \in \mathsf{Fn}(\mathcal{T})$, 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 α may contain more then one role r with $r \in \mathsf{Fn}(\mathcal{T})$. 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 $s \notin \mathsf{Fn}(\mathcal{T})$. 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 $s \notin \mathsf{Fn}(\mathcal{T})$ 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 $\mathcal{ALCOIF}^2_{reg}$ 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 $r_1, r_2 \in \mathsf{Fn}(\mathcal{T})$ distinct without traversing an $S \in \{s, s^-\}$ with $s \notin \mathsf{Fn}(\mathcal{T})$ in between. The sandwiched S allows us to jump and thus to move from the slice of the Σ-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. ❏

This finishes the proof of Theorem 18. It follows from our proof that every finitely satisfiable $\mathcal{SHOIF}^2_{reg}$ 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 $\mathcal{ALCOIF}^1_{reg}$ and $\mathcal{ALCOIF}^2_{reg}$. For example, we do not know how to accommodate local (unqualified) functionality restrictions, that is, concept expressions of the form $(\leqslant\ 1\ r)$. For qualified functionality restrictions $(\leqslant\ 1\, r\, C)$, it can be observed that this is indeed difficult. Let $\mathcal{ALCOIF}^1_{reg,qual}$ denote the extension of $\mathcal{ALCOIF}^1_{reg}$ with restrictions of the latter kind. It turns out that proving decidability of $\mathcal{ALCOIF}^1_{reg,qual}$ 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 $\mathcal{ALCOIF}^1_{reg,qual}$.*

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 μ-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.

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 μ-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 $\mathcal{SHOIQ}$ with transitive closure of roles. In *Proceedings of ISWC*, 264–279. Springer.

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 μ-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.

W3C OWL Working Group. 2009 (accessed July 2, 2020). *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation. Available at http://www.w3.org/TR/owl2-overview/.

Zeume, T., and Harwath, F. 2016. Order-invariance of two-variable logic is decidable. In *Proceedings of LICS*, 807–816.