TECHNISCHE UNIVERSITÄT DRESDEN

Technische Universität Dresden
Institute for Theoretical Computer Science
Chair for Automata Theory

# LTCS–Report

## Concept Descriptions with Set Constraints and Cardinality Constraints

Franz Baader

LTCS-Report 17-02

*The first version of this report was put online on April 6, 2017. The current version, containing more information on related work, was put online on July 13, 2017.*

*This is an extended version of a paper published in the proceedings of FroCoS 2017.*

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

# Contents

# Concept Descriptions with Set Constraints and Cardinality Constraints

Franz Baader
Theoretical Computer Science, TU Dresden
franz.baader@tu-dresden.de

**Abstract**

We introduce a new description logic that extends the well-known logic $\mathcal{ALCQ}$ by allowing the statement of constraints on role successors that are more general than the qualified number restrictions of $\mathcal{ALCQ}$. To formulate these constraints, we use the quantifier-free fragment of Boolean Algebra with Presburger Arithmetic (QFBAPA), in which one can express Boolean combinations of set constraints and numerical constraints on the cardinalities of sets. Though our new logic is considerably more expressive than $\mathcal{ALCQ}$, we are able to show that the complexity of reasoning in it is the same as in $\mathcal{ALCQ}$, both without and with TBoxes.

## 1 Introduction

Description Logics (DLs) [1] are a well-investigated family of logic-based knowledge representation languages, which are frequently used to formalize ontologies for application domains such as biology and medicine [10]. To define the important notions of such an application domain as formal concepts, DLs state necessary and sufficient conditions for an individual to belong to a concept. These conditions can be Boolean combinations of atomic properties required for the individual (expressed by concept names) or properties that refer to relationships with other individuals and their properties (expressed as role restrictions). For example, the concept of a man (i.e., a non-female human) that has a wife and only daughters can be formalized by the concept description

$$\mathit{Human} \sqcap \neg \mathit{Female} \sqcap \exists \mathit{spouse}.\mathit{Female} \sqcap \forall \mathit{child}.\mathit{Female},$$

which uses the concept names *Human* and *Female* and the role names *spouse* and *child* as well as the concept constructors conjunction ($\sqcap$), negation ($\neg$), value restriction ($\forall r.C$), and existential restriction ($\exists r.C$). Number restrictions

can express to how many individuals, possibly with certain properties, an element of the concept is related to for a given role. For example, the concept of a woman that has two daughters, three sons, and no other children can be formalized as

$$\mathit{Human} \sqcap \mathit{Female} \sqcap (\geqslant 2\ \mathit{child}.\mathit{Female}) \sqcap (\geqslant 3\ \mathit{child}.\neg\mathit{Female}) \sqcap (\leqslant 5\ \mathit{child}).$$

The first two number restrictions in this concept description are called *qualified* since they restrict the number of role successors belonging to certain concepts, whereas the last number restriction is *unqualified* since it is concerned with all role successors. Number restrictions have been used as concept constructors for DLs for a long time, but first only in the unqualified variant [3, 12]. Qualified number restrictions were first introduced and investigated in [11], but it took almost a decade before the exact complexity of reasoning in the DL $\mathcal{ALCQ}$, which has all the concept constructors introduced in the above examples, could be determined [19]. In fact, the tableau-algorithm for deciding the satisfiability of an $\mathcal{ALCQ}$ concept described in [11] generates $n$ new individuals to satisfy a qualified at-least restriction $\geqslant n\, r.C$. If we assume binary rather than unary representation of numbers (i.e., the size of $n$ in a number restriction is assumed to be $\log n$ rather than $n$), then this clearly generates exponentially many individuals, and thus the algorithm needs exponential space. The PSpace algorithm described in [19] does not keep $n$ successors in memory at the same time. Instead, it uses appropriate book-keeping of the number of successors (represented in binary) and comparisons of numbers to determine a clash between at-least and at-most restrictions. In order to improve the performance of reasoners for DLs with qualified number restrictions, also more sophisticated numerical reasoning approaches (such as linear integer programming) have been employed (see, e.g., [9, 6, 8]).

More expressive number restrictions have been introduced in [2]. On the one hand, that paper considers number restrictions on complex roles, i.e., roles that are constructed from role names using operations on binary relations such as intersection and composition. For example, using role composition within a number restriction, one can express that someone has at least four grandchildren:

$$\mathit{Human} \sqcap (\geqslant 4\ \mathit{child} \circ \mathit{child}).$$

One the other hand, the paper introduces symbolic number restrictions, in which variables can be used in place of explicit numbers. This allows one to express, e.g., that someone has more daughters than sons without specifying the actual number of them:

$$\mathit{Human} \sqcap \downarrow\alpha((\geqslant \alpha\ \mathit{child}.\mathit{Female}) \sqcap \neg(\geqslant \alpha\ \mathit{child}.\neg\mathit{Female})),$$

where $\downarrow\alpha$ says that there must exist such a cardinality $\alpha$. Unfortunately, both extensions on their own already lead to undecidability of reasoning if they are added to a DL that is closed under all Boolean operations.

In the present paper, we propose a new DL strictly extending $\mathcal{ALCQ}$, which we call $\mathcal{ALCSCC}$.[1] Among other things, this DL can describe some of the concepts expressible in the DLs introduced [2], but not in $\mathcal{ALCQ}$. Nevertheless, reasoning in our new DL is not only decidable, but of the same complexity as reasoning in $\mathcal{ALCQ}$. The basic idea underlying the definition of this logic is the following. A DL concept expresses under what conditions an individual $d$ belongs to the concept. On the one hand, these conditions refer to concept names to which $d$ must or must not belong. On the other hand, they state conditions on the individuals that are related to $d$ via some role. For example, the value restrictions $\forall r.C$ says that the set of $r$-successors of $d$ is contained in the set of elements of $C$. Thus, such a value restriction states an inclusion constraint between sets. Number restrictions enforce cardinality constraints on sets. For example, the qualified number restriction $\geqslant n\, r.C$ says that the cardinality of the set obtained by intersecting the set of $r$-successors of $d$ with the set of elements of $C$ has cardinality at least $n$. We now integrate into our DL a logic that can express set constraints (such as inclusion constraints) and numerical constraints regarding the cardinality of sets. This logic is called QFBAPA, which stands for the quantifier-free fragment of Boolean Algebra with Presburger Arithmetic. Basically, the Boolean algebra part of this logic can be used to build set expressions and the Presburger arithmetic part can state numerical constraints. Both parts are linked by the cardinality function. It has been shown in [13] that satisfiability of QFBAPA formulae is an NP-complete problem. Our PSpace algorithm for deciding the satisfiability of $\mathcal{ALCSCC}$ concept descriptions (see Section 5) and our ExpTime algorithm for deciding satisfiability in $\mathcal{ALCSCC}$ w.r.t. TBoxes (see Section 6) use the NP decision procedure for satisfiability of QFBAPA formulae as subprocedure.

Ohlbach and Koehler [14] have introduced a DL that also allows for Boolean set terms and arithmetic constraints on the cardinality of role successors. The expressiveness of their logic is somewhat different from ours (see Section 7). The major difference to our work is, however, that Ohlbach and Koehler give only decidability results and no complexity results. In addition, they only consider satisfiability of concept descriptions, whereas we also consider satisfiability w.r.t. TBoxes consisting of general concept inclusions (GCIs). In fact, we show in Section 6 that also w.r.t. GCIs the complexity of the satisfiability problem in $\mathcal{ALCSCC}$ is the same as in $\mathcal{ALCQ}$, i.e., ExpTime complete. Demri and Lugiez [5] introduce the modal logic EML, which allows for arithmetic constraints on the cardinality of successors w.r.t. the transition relations `R` as well as for automata-based constraints describing which formulae the ordered collection of `R`-successors of a given world must satisfy. They show that the satisfiability problem for EML is PSpace-complete. Since it is easy to see that $\mathcal{ALCSCC}$ is a syntactic variant of EML without automata-based constraints, this yields an alternative proof for the

[1] The name $\mathcal{ALCSCC}$ for our new DL is supposed to indicate that it extends the basic DL $\mathcal{ALC}$ with set and cardinality constraints rather than just qualified number restrictions.

fact that satisfiability of $\mathcal{ALCSCC}$ concept descriptions is in PSpace.[2] Demri and Lugiez do *not* show a result corresponding to our ExpTime result for satisfiability w.r.t. TBoxes in $\mathcal{ALCSCC}$.

# 2 Preliminaries

Before defining $\mathcal{ALCSCC}$ in Section 3, we briefly introduce $\mathcal{ALCQ}$ and QFBAPA.

Given disjoint finite sets $N_C$ and $N_R$ of concept names and role names, respectively, the set of $\mathcal{ALCQ}$ concept descriptions is defined inductively:

- all concept names are $\mathcal{ALCQ}$ concept descriptions;
- if $C, D$ are $\mathcal{ALCQ}$ concept descriptions, $r \in N_R$, and $n$ is a non-negative integer, then $\neg C$ (negation), $C \sqcup D$ (disjunction), $C \sqcap D$ (conjunction), $\geqslant n\, r.C$ and $\leqslant n\, r.C$ (qualified number restrictions) are $\mathcal{ALCQ}$ concept descriptions.

An $\mathcal{ALCQ}$ GCI is of the form $C \sqsubseteq D$ where $C, D$ are $\mathcal{ALCQ}$ concept descriptions. An $\mathcal{ALCQ}$ TBox is a finite set of $\mathcal{ALCQ}$ GCIs.

The semantics of $\mathcal{ALCQ}$ is defined using the notion of an interpretation. An *interpretation* is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where the *domain* $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function that assigns to every concept name $A$ a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to every role name $r$ a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. This function is extended to $\mathcal{ALC}$-concept descriptions as follows:

- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}, (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}, (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$;
- $(\geqslant n\, r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{there are at least } n\ y \in \Delta^{\mathcal{I}} \text{ with } (x, y) \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$;
- $(\leqslant n\, r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{there are at most } n\ y \in \Delta^{\mathcal{I}} \text{ with } (x, y) \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$.

The interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ if it satisfies $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all GCIs $C \sqsubseteq D \in \mathcal{T}$. Given an $\mathcal{ALCQ}$ concept description $C$, we say that $C$ is *satisfiable* if there is an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$. Analogously, $C$ is *satisfiable w.r.t. the TBox* $\mathcal{T}$ if there is a *model* $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$. Two $\mathcal{ALCQ}$ concept descriptions $C, D$ are *equivalent* (written $C \equiv D$) if $C^{\mathcal{I}} = D^{\mathcal{I}}$ holds for all interpretations $\mathcal{I}$. Other inference problems such as subsumption can be reduced to satisfiability, which is why we concentrate on it. The introduced notions (GCI, TBox, model, satisfiability, and equivalence) can of course also be used for DLs

[2] The author of the present paper learned about the PSpace result in [5] only after publishing his complexity results for $\mathcal{ALCSCC}$ at the conference FroCoS 2017.

other than $\mathcal{ALCQ}$, and in particular for the DL $\mathcal{ALCSCC}$ introduced in the next section.

The DL $\mathcal{ALC}$ differs from $\mathcal{ALCQ}$ in that it has existential restrictions ($\exists r.C$) and value restrictions ($\forall r.C$) as constructors in place of qualified number restrictions. It is a sublogic of $\mathcal{ALCQ}$ since these two constructors can be expressed using qualified number restrictions: $\exists r.C \equiv \geqslant 1\, r.C$ and $\forall r.C \equiv \leqslant 0\, r.\neg C$.

Let us now briefly introduce the logic *QFBAPA* (more details can be found in [13]). In this logic one can build *set terms* by applying Boolean operations (intersection, union, and complement) to set variables as well as the constants $\emptyset$ and $\mathcal{U}$. Set terms $s, t$ can the be used to state inclusion and equality constraints ($s = t, s \subseteq t$) between sets. *Presburger Arithmetic (PA) expressions* are built from integer variables, integer constants, and set cardinalities $|s|$ using addition as well as multiplication with an integer constant. They can be used to form numerical constraints of the form $k = \ell, k < \ell, N \,\mathsf{dvd}\, \ell$, where $k, \ell$ are PA expressions, $N$ is an integer constant, and $\mathsf{dvd}$ stands for divisibility. A *QFBAPA formula* is a Boolean combination of set and numerical constraints.

A *solution* $\sigma$ of a QFBAPA formula $\phi$ assigns a finite set $\sigma(\mathcal{U})$ to $\mathcal{U}$, subsets of $\sigma(\mathcal{U})$ to set variables, and integers to integer variables such that $\phi$ is satisfied by this assignment. The evaluation of set terms, PA expressions, and set and numerical constraints w.r.t. $\sigma$ is defined in the obvious way. For example, $\sigma$ satisfies the numerical constraint $|s \cup t| = |s| + |t|$ for set variables $s, t$ if the cardinality of the union of the sets $\sigma(s)$ and $\sigma(t)$ is the same as the sum of the cardinalities of these sets. Note that this is the case iff $\sigma(s)$ and $\sigma(t)$ are disjoint, which we could also have expressed using the set constraint $s \cap t \subseteq \emptyset$. A QFBAPA formula $\phi$ is *satisfiable* if it has a solution.

# 3 Syntax and semantics of $\mathcal{ALCSCC}$

Basically, the DL $\mathcal{ALCSCC}$ has all Boolean operations as concept constructors and can state constraints on role successors using the expressiveness of QFBAPA.

Given a finite set of set symbols $T$ with $\{\emptyset, \mathcal{U}\} \cap T = \emptyset$, *set terms* over $T$ are defined inductively as follows:

- the symbols $\emptyset$ and $\mathcal{U}$ are set terms;
- every set symbol is a set term;
- if $s, t$ are set terms, then so are $s \cup t$, $s \cap t$, and $s^c$.

*Cardinality terms* over $T$ are also defined inductively:[3]

[3] In contrast to PA expressions, we do not have integer variables here and numerical constants

- every non-negative integer $N$ is a cardinality term;
- if $s$ is a set term, then $|s|$ is a cardinality term;
- if $k, \ell$ are cardinality terms, then so are $k+\ell$ and $N \cdot \ell$ for every non-negative integer $N$.

*Set constraints* over $T$ are of the form $s = t, s \subseteq t$ or their negation for set terms $s, t$. *Cardinality constraints* over $T$ are of the form $k = \ell$, $k < \ell$, $k \leq \ell$, $N \,\mathsf{dvd}\, \ell$ or their negation for cardinality terms $k, \ell$ and a non-negative integer $N > 0$.

Given a set $\Delta^{\mathcal{I}}$ and a mapping $\cdot^{\mathcal{I}}$ that maps

- $\emptyset$ to $\emptyset^{\mathcal{I}} = \emptyset$,
- $\mathcal{U}$ to a *finite* subset $\mathcal{U}^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, and
- every symbol $\sigma$ in $T$ to a subset $\sigma^{\mathcal{I}}$ of $\mathcal{U}^{\mathcal{I}}$,

we extend this mapping to set terms and cardinality terms as follows:

- $(s \cup t)^{\mathcal{I}} = s^{\mathcal{I}} \cup t^{\mathcal{I}}$, $(s \cap t)^{\mathcal{I}} = s^{\mathcal{I}} \cap t^{\mathcal{I}}$, and $(s^c)^{\mathcal{I}} = \mathcal{U}^{\mathcal{I}} \setminus s^{\mathcal{I}}$,
- $|s|^{\mathcal{I}} = |s^{\mathcal{I}}|$,
- $(k + \ell)^{\mathcal{I}} = k^{\mathcal{I}} + \ell^{\mathcal{I}}$ and $(N \cdot \ell)^{\mathcal{I}} = N \cdot \ell^{\mathcal{I}}$.

This mapping satisfies

- the set constraint $s = t$ if $s^{\mathcal{I}} = t^{\mathcal{I}}$, and its negation if $s^{\mathcal{I}} \neq t^{\mathcal{I}}$,
- the set constraint $s \subseteq t$ if $s^{\mathcal{I}} \subseteq t^{\mathcal{I}}$, and its negation if $s^{\mathcal{I}} \not\subseteq t^{\mathcal{I}}$,
- the cardinality constraint $k = \ell$ if $k^{\mathcal{I}} = \ell^{\mathcal{I}}$, and its negation if $k^{\mathcal{I}} \neq \ell^{\mathcal{I}}$,
- the cardinality constraint $k < \ell$ if $k^{\mathcal{I}} < \ell^{\mathcal{I}}$, and its negation if $k^{\mathcal{I}} \geq \ell^{\mathcal{I}}$,
- the cardinality constraint $k \leq \ell$ if $k^{\mathcal{I}} \leq \ell^{\mathcal{I}}$, and its negation if $k^{\mathcal{I}} > \ell^{\mathcal{I}}$,
- the cardinality constraint $N \,\mathsf{dvd}\, \ell$ if there is a non-negative integer $M$ such that $N \cdot M = \ell^{\mathcal{I}}$, and its negation if there is no such $M$.

Given disjoint *finite* sets $N_C$ and $N_R$ of concept names and role names, respectively, we define the set of $\mathcal{ALCSCC}$ concept descriptions by induction:

- every concept name is an $\mathcal{ALCSCC}$ concept description;

must be non-negative.

- if $C, D$ are $\mathcal{ALCSCC}$ concept descriptions, then so are $C \sqcap D, C \sqcup D, \neg C$;
- if $c$ is a set constraint or a cardinality constraint over as finite set of symbols consisting of role names and $\mathcal{ALCSCC}$ concept descriptions, then $succ(c)$ is an $\mathcal{ALCSCC}$ concept description.

As usual, we will use $\top$ (top) and $\bot$ (bottom) as abbreviations for $A \sqcup \neg A$ and $A \sqcap \neg A$, respectively.

An *interpretation* of $N_C$ and $N_R$ consists of a set non-empty $\Delta^{\mathcal{I}}$ and a mapping $\cdot^{\mathcal{I}}$ that maps

- every concept name $A \in N_C$ to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$;
- every role name $r \in N_R$ to a binary relation $r^{\mathcal{I}}$ over $\Delta^{\mathcal{I}}$ such that every element of $\Delta^{\mathcal{I}}$ has only finitely many $r$-successors, i.e., the set

  $$r^{\mathcal{I}}(d) := \{e \in \Delta^{\mathcal{I}} \mid (d, e) \in r^{\mathcal{I}}\}$$

  is finite for all $d \in \Delta^{\mathcal{I}}$.

The interpretation function $\cdot^{\mathcal{I}}$ is inductively extended to $\mathcal{ALCSCC}$ concept descriptions as follows:

- $(C \sqcup D)^{\mathcal{I}} := C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$, and $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$;
- $succ(c)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid \text{the mapping } \cdot^{\mathcal{I}_d} \text{ satisfies } c\}$,
  where $\cdot^{\mathcal{I}_d}$ maps $\emptyset$ to $\emptyset^{\mathcal{I}} = \emptyset$, $\mathcal{U}$ to $\mathcal{U}^{\mathcal{I}} = rs^{\mathcal{I}}(d)$, where

  $$rs^{\mathcal{I}}(d) := \bigcup_{r \in N_R} r^{\mathcal{I}}(d),$$

  and the concept descriptions and role names occurring in $c$ to subsets of $\mathcal{U}^{\mathcal{I}}$ as follows:

  $$C^{\mathcal{I}_d} := C^{\mathcal{I}} \cap rs^{\mathcal{I}}(d)$$

  for concept descriptions $C$ occurring in $c$ and $r^{\mathcal{I}_d} := r^{\mathcal{I}}(d)$.

Note that $\cdot^{\mathcal{I}_d}$ is well-defined since we can assume by induction that $C^{\mathcal{I}}$ is already defined for concept descriptions $C$ occurring in $c$. In addition, it indeed maps $\mathcal{U}$ to a *finite* set since $rs^{\mathcal{I}}(d)$ is finite due to the facts that (i) $N_R$ is finite, and (ii) every element of $\Delta^{\mathcal{I}}$ has only finitely many $r$-successors for all role names $r \in N_R$.

Also note that top and bottom are interpreted as the whole interpretation domain and the empty set, respectively, i.e. $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$.

# 4 Expressive power

We claim that $\mathcal{ALCSCC}$ has the description logic $\mathcal{ALCQ}$ [11, 19] as sublogic. For this it is sufficient to show that qualified number restrictions $\geqslant n\, r.C$ and $\leqslant n\, r.C$ can be expressed in $\mathcal{ALCSCC}$.

**Lemma 1** *For all interpretations $\mathcal{I}$ we have*

$$(\geqslant n\, r.C)^{\mathcal{I}} = succ(|C \cap r| \geq n)^{\mathcal{I}} \quad and \quad (\leqslant n\, r.C)^{\mathcal{I}} = succ(|C \cap r| \leq n)^{\mathcal{I}}.$$

As an easy consequence we obtain that reasoning (e.g., subsumption, satisfiability) in $\mathcal{ALCSCC}$ is at least as complex as reasoning in $\mathcal{ALCQ}$, i.e., PSpace-hard without a TBox and ExpTime-hard w.r.t. a general TBox. The only thing to take care of here is that the notion of interpretation defined above is more restrictive than the one used for $\mathcal{ALCQ}$ since in $\mathcal{ALCQ}$ individuals are not required to have only finitely many role successors. However, due to the fact that $\mathcal{ALCQ}$ has the finite model property, we can assume without loss of generality that interpretations of $\mathcal{ALCQ}$ satisfy the finite-role-successors property required in this paper for interpretations.

We can, however, express things in $\mathcal{ALCSCC}$ that cannot be expressed in $\mathcal{ALCQ}$.[4] For example, we can define the persons that have the same number of brothers as sisters by writing

$$Person \sqcap succ(|child \cap Male| = |child \cap Female|).$$

Description Logics that can express such restrictions have been introduced in [2], but due to the use of explicit variables for cardinalities of sets of role successors in the logic defined in [2], this logic becomes undecidable.

In [2], also number restrictions on complex role expressions are considered, but again the high expressiveness of the corresponding logics introduced in [2] often leads to undecidability. We can express weaker versions of such restrictions in $\mathcal{ALCSCC}$. For example,

$$Employer \sqcap succ(|related \cap employs| \leq 1)$$

describes employers that employ at most one relative, and

$$Employer \sqcap succ(2 \cdot |related \cap employs| < |employs|)$$

describes employers that employ more no-relatives than relatives. Using divisibility cardinality constraints, we can for example express creatures that have an even number of legs as

$$Creature \sqcap succ(2\, \mathsf{dvd}\, |has\text{-}limb \cap Leg|),$$

[4] To show in a formal way that these things cannot be expressed in $\mathcal{ALCQ}$ we could employ an appropriate notion of bisimulation for $\mathcal{ALCQ}$ [4].

without having to specify how man legs the respective creature actually has.

As an example for an inexpressibility proof in $\mathcal{ALCQ}$, we consider a simplified version of our first example.

**Lemma 2** *The $\mathcal{ALCSCC}$ concept description $succ(|r| = |s|)$ for distinct role names $r, s$ cannot be expressed in $\mathcal{ALCQ}$.*

*Proof.* Assume that $C$ is an $\mathcal{ALCQ}$ concept description such that, for all interpretations $\mathcal{I}$, we have $C^{\mathcal{I}} = succ(|r| = |s|)^{\mathcal{I}}$. Let $n$ be a non-negative integer that is larger than the largest number occurring in a number restriction in $C$. Consider an interpretation $\mathcal{I}$ with $\Delta^{\mathcal{I}} = \{0, 1, 2, \ldots\}$ such that

$$r^{\mathcal{I}} = \{(0, i) \mid 1 \leq i \leq n\} \text{ and } s^{\mathcal{I}} = \{(0, n + i) \mid 1 \leq i \leq n\}.$$

Then $0 \in succ(|r| = |s|)^{\mathcal{I}}$ and thus $0 \in C^{\mathcal{I}}$. We change $\mathcal{I}$ to $\mathcal{I}'$ by giving 0 an additional $s$-successor, i.e., $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{I}}$, $r^{\mathcal{I}'} = r^{\mathcal{I}}$, and $s^{\mathcal{I}'} = s^{\mathcal{I}} \cup \{(0, 2n + 1)\}$. Then $0 \notin succ(|r| = |s|)^{\mathcal{I}'}$. However, since all the numbers occurring in number restrictions in $C$ are smaller than $n$, changing the number of $s$-successors of 0 from $n$ to $n + 1$ has no impact on whether 0 belongs to $C$ or not. Consequently, we have $0 \in C^{\mathcal{I}'}$, and thus $C^{\mathcal{I}'} \neq succ(|r| = |s|)^{\mathcal{I}'}$, which yields a contradiction to our assumption that $C$ expresses $succ(|r| = |s|)$. □

# 5 Satisfiability of $\mathcal{ALCSCC}$ concept descriptions

Recall that the $\mathcal{ALCSCC}$ concept description $C$ is satisfiable if there is an interpretation $\mathcal{I}$ and an element $d \in \Delta^{\mathcal{I}}$ such that $d \in C^{\mathcal{I}}$. We call $\mathcal{I}$ a model of $C$ and $d$ a witness for the satisfaction of $C$ in $\mathcal{I}$.

Since $\mathcal{ALCSCC}$ can express $\mathcal{ALCQ}$ and thus also $\mathcal{ALC}$, the satisfiability problem for $\mathcal{ALCSCC}$ concept descriptions is PSpace-hard [18]. In this section, we use the ideas underlying the proof that satisfiability in QFBAPA is in NP [13] to show a matching upper bound (assuming binary representation of numbers). For $\mathcal{ALCQ}$ such an upper bound was first shown in [19].

A given $\mathcal{ALCSCC}$ concept description is a Boolean combination of *atoms*, i.e., concept names $A$ and successor constraints $succ(c)$ for set or cardinality constraints $c$. Viewing these atoms as propositional variables, we first guess which of them are true and which are false. In case the guessed assignment does not satisfy the propositional formula corresponding to $C$, we fail. Otherwise, the assignment tells us that there is a way to assign concept names to an individual such that the part of $C$ that concerns atoms that are concept names is satisfied. It remains to see whether such an individual can receive role successors such that the part of $C$ that concerns atoms that are successors constraints can be satisfied as well.

Before showing how this can be done in general, let us consider a simple example.

**Example 3** Let

$$C := (\neg A \sqcup \neg succ(2 \,\mathsf{dvd}\, |r|)) \sqcap (\neg B \sqcup succ(|r| = 2 \cdot |s|)).$$

If we guess that the atoms $A$ and $B$ should be true, then we need to guess that the atom $succ(2 \,\mathsf{dvd}\, |r|)$ is false and the atom $succ(|r| = 2 \cdot |s|)$ is true since otherwise the propositional formula corresponding to $C$ would become false, leading to failure. Consequently, we need an individual that belongs to $A$ and $B$ and whose role successors satisfy the constraints $\neg(2 \,\mathsf{dvd}\, |r|)$ and $|r| = 2 \cdot |s|$. If we replace the concept names $r$ and $s$ in these constraints by set variables $X_r$ and $X_s$, respectively, then we obtain the QFBAPA formula $\neg(2 \,\mathsf{dvd}\, |X_r|) \wedge |X_r| = 2 \cdot |X_s|$. Obviously, this formula is not satisfiable since the second conjunct requires $|X_r|$ to be even, whereas the first one forbids this.
Now assume that we have guessed that the atom $A$ is false and the atoms $B$, $succ(2 \,\mathsf{dvd}\, |r|)$, and $succ(|r| = 2 \cdot |s|)$ are true. This yields the QFBAPA formula $2 \,\mathsf{dvd}\, |X_r| \wedge |X_r| = 2 \cdot |X_s|$, which can be satisfied by assigning the set $\{d_1, d_2\}$ to $X_r$ and the set $\{d_2\}$ to $X_s$. Thus, if we build the interpretation $\mathcal{I}$ with domain $\{d_0, d_1, d_2\}$ where $d_0$ belongs to $B$, but not to $A$, and where $d_1, d_2$ are the $r^{\mathcal{I}}$-successors of $d_0$ and $d_2$ is the only $s^{\mathcal{I}}$-successors of $d_0$, then we have $d_0 \in C^{\mathcal{I}}$.

When building the QFBAPA formula corresponding to an assignment, we need to take the semantics of $\mathcal{ALCSCC}$ into account, which says that, when evaluating the successors constraints of a given individual $d$, the set $\mathcal{U}$ must consist of exactly the role successors of this individual. Consequently, in addition to the conjuncts induced by the successor constraints on the top-level of $C$, the QFBAPA formula must contain the conjunct

$$X_{r_1} \cup \ldots \cup X_{r_n} = \mathcal{U},$$

where $N_R = \{r_1, \ldots, r_n\}$. In the above example, the presence of this conjunct is irrelevant. The following example shows why it is in general necessary to add this conjunct.

**Example 4** Let

$$C := succ(|\mathcal{U}| \geq 1) \sqcap succ(r \subseteq \emptyset) \sqcap succ(|s| = 0),$$

where $N_R = \{r, s\}$. Then $C$ is unsatisfiable according to our semantics, but the QFBAPA formula $|\mathcal{U}| \geq 1 \wedge X_r \subseteq \emptyset \wedge |X_s| = 0$ is satisfiable. However, this QFBAPA formula becomes unsatisfiable if we add the conjunct $X_r \cup X_s = \mathcal{U}$.

Until now, we have considered examples where the successor constraints do not contain (possibly complex) concept descriptions. If this is the case, an additional problem needs to be solved, as illustrated by the next example, which is obtained by modifying Example 3.

**Example 5** Let

$$C := (\neg A \sqcup \neg succ(2 \,\mathsf{dvd}\, |D|)) \sqcap (\neg B \sqcup succ(|D| = 2 \cdot |E|)),$$

where $D, E$ are (possibly complex) $\mathcal{ALCSCC}$ concept descriptions. Guessing that the atom $A$ is false and the atoms $B$, $succ(2 \,\mathsf{dvd}\, |D|)$, and $succ(|D| = 2 \cdot |E|)$ are true, we obtain the QFBAPA formula $2 \,\mathsf{dvd}\, |X_D| \wedge |X_D| = 2 \cdot |X_E| \wedge \bigcup_{r \in N_R} X_r = \mathcal{U}$. One solution of this formula is the one that assigns $\{d_1, d_2\}$ to $X_D$, $\{d_2\}$ to $X_E$, and $\{d_1, d_2\}$ to all the variables $X_r$ for $r \in N_R$.
In contrast to the case considered in Example 3, the existence of such a solution does not yet show that $C$ is satisfiable. In fact, this solution requires $d_1$ to belong to $D$, but not to $E$, whereas $d_2$ must belong to both $D$ and $E$. This is only possible if the concept descriptions $D \sqcap \neg E$ and $D \sqcap E$ are satisfiable. Thus, we need recursive calls of the satisfiability procedures for $\mathcal{ALCSCC}$ for these two inputs. This recursion is well-founded (with a linear recursion depth) since the nesting depth of successor constraints in $D$ and $E$ (and thus in $D \sqcap \neg E$ and $D \sqcap E$) is by at least one smaller than the nesting depth in $C$.
Now assume that these recursive calls yield the result that $D \sqcap \neg E$ is satisfiable, but $D \sqcap E$ is not. This does not mean that $C$ is unsatisfiable. In fact, there is also a solution of the above QFBAPA formula that assigns $\{d_1, d_2\}$ to $X_D$, $\{d_3\}$ to $X_E$, and $\{d_1, d_2, d_3\}$ to all the variables $X_r$ for $r \in N_R$. This solution requires $D \sqcap \neg E$ and $\neg D \sqcap E$ to be satisfiable. Assuming that this is the case also for the latter concept description, we can construct an interpretation $\mathcal{I}$ containing an element $d_0$ that has the individuals $d_1, d_2, d_3$ as role successors for all roles $r \in N_R$. The rest of $\mathcal{I}$ is a disjoint union of two models of $D \sqcap \neg E$ with a model of $\neg D \sqcap E$, where the respective witnesses are identified with $d_1, d_2$, and $d_3$. By construction, this yields a model of $C$ with witness $d_0$.

Summing up, we have illustrated by the above examples that a guessed assignment for the top-level atoms of $C$ either leads to failure (if the propositional formula corresponding to $C$ is not satisfied by the assignment) or it yields a QFBAPA formula corresponding to the successor constraints under this assignment. Unsatisfiability of this QFBAPA formula again leads to failure. A solution for the QFBAPA formula creates recursive calls of the satisfiability procedure, where the inputs have a smaller nesting depth of successor constraints than $C$. In case one of these recursive calls returns "unsatisfiable," we cannot conclude that $C$ is unsatisfiable. In fact, it may be the case that another solution of the QFBAPA formula creates other recursive calls, which may all yield "satisfiable." The remaining question is now how to find such a solution in case one exists.

A naive idea could be to add the information that a certain combination of concepts (i.e., a conjunction of concepts and negated concepts) is unsatisfiable to the QFBAPA formula. In Example 5, after finding out that $D \sqcap E$ is unsatisfiable, we could have added the conjunct $|X_D \cap X_E| = 0$ to ensure that the next solution does not require $D \sqcap E$ to be satisfiable. The problem with this approach

is that the next solution may create another recursive call returning "unsatisfiable," and thus an additional conjunct needs to be added (e.g., if $\neg D \sqcap \neg E$ turns out to be unsatisfiable, we need to add $|X_D^c \cap X_E^c| = 0$), etc. If the top-level successor constraints of $C$ contain $k$ concept descriptions, then in the worst case a number of conjuncts that is exponential in $k$ may need to be added to the QFBAPA formula. Since satisfiability of QFBAPA formulae is NP-complete, testing the resulting exponentially large QFBAPA formula for satisfiability would require non-deterministic exponential time and representing the formula would need exponential space.

In order to stay within PSpace, we use a result from [13], which is the main tool used there to show that satisfiability in QFBAPA is in NP. Assume that $\phi$ is a QFBAPA formula containing the set variables $X_1, \ldots, X_k$. A *Venn region* is of the form

$$X_1^{c_1} \cap \ldots \cap X_k^{c_k},$$

where $c_i$ is either empty or $c$ for $i = 1, \ldots, k$. It is shown in [13] that, given $\phi$, one can easily compute a number $N$ whose value is polynomial in the size of $\phi$ such that the following holds: $\phi$ is satisfiable iff it has a solution in which $\leq N$ Venn regions are interpreted by non-empty sets. Taking a closer look at how this result is proved in [13], one can actually strengthen it.

**Lemma 6** *For every QFBAPA formula $\phi$, one can compute in polynomial time a number $N$ whose value is polynomial in the size of $\phi$ such that the following holds for every solution $\sigma$ of $\phi$: there is a solution $\sigma'$ of $\phi$ such that*

- $|\{v \mid v \text{ Venn region and } \sigma'(v) \neq \emptyset\}| \leq N$, *and*
- $\{v \mid v \text{ Venn region and } \sigma'(v) \neq \emptyset\} \subseteq \{v \mid v \text{ Venn region and } \sigma(v) \neq \emptyset\}$.

*Proof.* Consider the argument in Section 3 of [13] below Fact 1, and use as set $X$ the one consisting of all the vectors $x_\beta$ corresponding to Venn regions $\beta$ non-empty w.r.t. $\sigma$ rather than using all possible Venn regions. Applying Fact 1 in [13] to this set $X$ yields an appropriate number $N$ and a subset $\tilde{X} \subseteq X$ of cardinality $\leq N$. The elements of $\tilde{X}$ correspond to the non-empty Venn regions in the new solution $\sigma'$, which assigns to all set expressions in $\phi$ the same cadinalities as $\sigma$. The inclusion of $\tilde{X}$ in $X$ yields the desired inclusion relation between non-empty Venn regions w.r.t. $\sigma'$ and $\sigma$, respectively. □

We can now continue with the description of our approach. Given a QFBAPA formula $\phi$ induced by our assignment for the top-level atoms of $C$, we compute the corresponding number $N$ and then guess $\leq N$ Venn regions to be interpreted as non-empty sets. For each of these Venn regions $X_1^{c_1} \cap \ldots \cap X_k^{c_k}$, we add the conjunct $|X_1^{c_1} \cap \ldots \cap X_k^{c_k}| \geq 1$ to $\phi$. In addition, we add the conjunct that states that the union of the guessed Venn regions is equal to $\mathcal{U}$, and thus that all other

Venn regions are empty. The resulting QFBAPA formula $\psi$ has a size that is polynomial in the size of $\phi$, and thus of $C$. We then

1. test whether $\psi$ is satisfiable using the NP satisfiability algorithm for QFBAPA;
2. for every guessed Venn region, we consider the part that consists of set variables corresponding to concept descriptions, and recursively test the induced concept descriptions for satisfiability.

If $\phi$ is satisfiable, then there is a solution in which $\leq N$ Venn regions are interpreted by non-empty sets, and thus the first test is successful for one of the guessed sets of Venn regions. Due to the construction of $\psi$, the corresponding solution interprets all other Venn regions as empty sets. Consequently, it is sufficient to test the concept descriptions considered in 2. for satisfiability. If all tests are successful then we can construct a model of $C$ as illustrated in Example 5. Basically, this model has a witness $d_0$ whose role successors w.r.t. all roles in $N_R$ are determined by the solutions for the set variables corresponding to roles. These successors are witnesses for the concept descriptions considered in 2., where the respective models are made disjoint and reproduced as many times as needed.

**Theorem 7** *Satisfiability of $\mathcal{ALCSCC}$ concept descriptions is PSpace-complete.*

*Proof.* Given an $\mathcal{ALCSCC}$ concept description $C$, the algorithm sketched above proceeds as follows:

1. It views the atoms (concept names and successor constraints) on the top level of $C$ (i.e., atoms that are not nested within successor constraints) as propositional variables, guesses a truth assignment for these variables, and then checks whether this assignment satisfies the propositional formula corresponding to $C$ (where the atoms are replaced by propositional variables). If this test is negative, then this run of the algorithm *fails*. Otherwise, it continues with the next step.
2. The truth assignment for the variables corresponding to successor constraints induces a QFBAPA formula $\phi$, as described above. We conjoin to this formula the set constraint $X_{r_1} \cup \ldots \cup X_{r_n} = \mathcal{U}$, where $N_R = \{r_1, \ldots, r_n\}$. For the resulting formula $\phi'$, we compute the number $N$ that bounds the number of Venn regions that need to be non-empty in a solution of $\phi'$ (see Lemma 6). Then we guess $\leq N$ Venn regions. For each of these Venn regions $X_1^{c_1} \cap \ldots \cap X_k^{c_k}$, we add the conjunct $|X_1^{c_1} \cap \ldots \cap X_k^{c_k}| \geq 1$ to $\phi'$. In addition, we add the conjunct that states that the union of the guessed Venn regions is equal to $\mathcal{U}$. For the resulting formula $\psi$, we test whether $\psi$ is satisfiable using the NP satisfiability algorithm for QFBAPA. If this

test is negative, then this run of the algorithm *fails*. Otherwise, it continues with the next step

3. For every guessed Venn region $v$, we consider the part that consists of set variables $X_D$ corresponding to concept descriptions $D$. We then build a concept description $C_v$ that contains a conjunct for every set variables $X_D$ occurring in $v$, where this conjunct is $D$ in case $v$ contains $X_D$ and it is $\neg D$ in case $v$ contains $X_D^c$. We then apply the algorithm recursively to $C_v$ for each of the guessed Venn regions $v$. If one of these applications *fails*, then this run of the algorithm *fails*. Otherwise, this run of the algorithm *succeeds*.

This algorithm indeed runs *in PSpace* since

- guessing is harmless due to Savitch's theorem, which says that PSpace is equal to NPSpace [7];
- the recursion stack for the recursive calls has linear depth since the nesting of successor restrictions decreases with each call, and for each concept to be tested, only polynomially many such calls are creates (since the values of the numbers $N$ are polynomial in the size of the tested concepts);
- the satisfiability test for QFBAPA formulae is in NP and applied to formulae of polynomial size.

Regarding *soundness* (i.e., if the algorithm succeeds, then the input concept $C$ is indeed satisfiable), we have already sketched above how a model of $C$ can be obtained from a successful run. Indeed, if Step 1 of the algorithm succeeds, then we create a witness $d_0$. The truth assignment for the propositional variables corresponding to concept names tells us, for every concept name $A$, whether $d_0$ needs to belong to $A$ or not. Regarding the role successors of $d_0$, we consider the solution for the QFBAPA formula $\psi$ found in Step 2 of the algorithm. Assume that this solution assigns the finite set $\{d_1, \ldots, d_m\}$ to the set term $\mathcal{U}$. Then $d_0$ receives the role successors $d_1, \ldots, d_m$, where the assignments for the set variables $X_r$ for $r \in N_R$ tell us which roles connect $d_0$ with these new individuals. Finally, each $d_i$ belongs to one of the guessed non-empty Venn regions $v$, and the recursive call of the algorithm with input $C_v$ was successful. By induction, we can assume that this implies the existence of a model $\mathcal{I}_v$ of $C_v$ with a witness $e_v$. We create a disjoint copy of $\mathcal{I}_v$ where the witness is replaced by $d_i$. Our interpretation $\mathcal{I}$ consists of the disjoint union of these copies, for $i = 1, \ldots, m$, together with $d_0$, where $d_0$ is linked by roles to the witnesses $d_1, \ldots, d_m$ as described above. A simple induction proof over the nesting depth of successor restrictions in $C$ can be used to show that $\mathcal{I}$ is a model of $C$ with witness $d_0$.

To show *completeness* (i.e., if $C$ is satisfiable, then the algorithm succeeds), assume that $\mathcal{I}$ is a model of $C$ with witness $d_0$. Then the membership and non-membership of $d_0$ in the top-level atoms of $C$ provides us with a truth assignment that satisfies the propositional formula corresponding to $C$. Thus, the first step of the algorithm succeeds if we guess this assignment. Let $d_1, \ldots, d_m$ be the finitely many role successors of $d_0$ in $\mathcal{I}$. We can use the membership of these successors in $r^{\mathcal{I}}(d_0)$ for $r \in N_R$ and in $D^{\mathcal{I}}$ for concept descriptions $D$ occurring in successor restrictions on the top-level of $C$ to obtain assignments of subsets of $\{d_1, \ldots, d_m\}$ to the set variables $X_r$ and $X_D$. The fact that $d_0 \in C^{\mathcal{I}}$ implies that the resulting assignment is a solution of the QFBAPA formula $\phi'$ constructed in Step 2 of the algorithm. However, this solution is not necessarily a solution of one of the formulae $\psi$ extending $\phi'$ corresponding to the guesses of $\leq N$ non-empty Venn regions. In fact, the assignment induced by $\mathcal{I}$ may make more than $N$ Venn regions non-empty. In this case, it cannot solve any of the formulae $\psi$ constructed in Step 2 of the algorithm. However, since $\phi'$ is solvable, by Lemma 6 it also has a solution that (i) makes $\leq N$ Venn regions non-empty, and (ii) only makes Venn regions non-empty that are also non-empty w.r.t. the solution induced by $\mathcal{I}$. Thus, we can guess the set of Venn regions that are non-empty in such a solution. This ensures that the corresponding formula $\psi$ has a solution. Because of (ii), each of the guessed Venn regions $v$ has a satisfiable concept $C_v$ since these Venn regions (and the corresponding concepts) are actually populated by one of the elements $d_1, \ldots, d_m$ of $\mathcal{I}$. □

# 6 Satisfiability in $\mathcal{ALCSCC}$ w.r.t. GCIs

Recall that the $\mathcal{ALCSCC}$ concept description $C$ is satisfiable w.r.t. a TBox $\mathcal{T}$ if there is a model $\mathcal{I}$ of $\mathcal{T}$ and an element $d \in \Delta^{\mathcal{I}}$ such that $d \in C^{\mathcal{I}}$. We call $\mathcal{I}$ a model of $C$ w.r.t. $\mathcal{T}$ and $d$ a witness for the satisfaction of $C$ w.r.t. $\mathcal{T}$ in $\mathcal{I}$. ExpTime-hardness of satisfiability in $\mathcal{ALCSCC}$ w.r.t. a TBox is an obvious consequence of the fact that satisfiability w.r.t. a TBox in the sublogic $\mathcal{ALC}$ of $\mathcal{ALCSCC}$ is already ExpTime-complete [17]. Thus, it is sufficient to show that satisfiability w.r.t. a TBox can be decided using only exponential time.

It is well-known that one can assume without loss of generality that the TBox consists of a single GCI of the form $\top \sqsubseteq D$. In fact, the TBox $\{C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n\}$ has obviously the same models as the TBox $\{\top \sqsubseteq (\neg C_1 \sqcup D_1) \sqcap \ldots \sqcap (\neg C_n \sqcup D_n)\}$. Thus, in the following we assume that $C_0$ is an $\mathcal{ALCSCC}$ concept description and $\mathcal{T} = \{\top \sqsubseteq D_0\}$ an $\mathcal{ALCSCC}$ TBox. We want to test whether $C_0$ is satisfiable w.r.t. $\mathcal{T}$.

A simple approach for showing that the satisfiability problem w.r.t. a TBox in a given DL is in ExpTime is *type elimination* [15, 16]. Basically, given a set of concept descriptions $\mathcal{S}$, the *type* of an individual in an interpretation consists

of the elements of $\mathcal{S}$ to which the individual belongs. If the set $\mathcal{S}$ contains the concept description $D_0$, then the type of any individual in a model of $\mathcal{T}$ must contain $D_0$. In addition, any witness for the satisfaction of $C_0$ w.r.t. $\mathcal{T}$ must contain $C_0$ in its type. Finally successor constraints occurring in the type of an individual imply that there exist other individuals whose types satisfy these constraints. For example, if there is an individual whose type contains the constraint $succ(|r \cap C| > 0)$, which corresponds to the existential restriction $\exists r.C$, then there must be an individual in the interpretation whose type contains $C$. Type elimination tries to find a collection of types that are exactly the types of a model $\mathcal{I}$ of $C_0$ w.r.t. $\mathcal{T}$ by starting with all possible types and eliminating those that contain successor constraints that cannot be satisfied by the still available types. For this to work correctly, the set $\mathcal{S}$ must contain sufficiently many concept descriptions. We assume in the following, that $\mathcal{S}$ contains *all subdescriptions* of $C_0$ and $D_0$ as well as the *negations of these subdescriptions*.

**Definition 8** A subset $t$ of $\mathcal{S}$ is a *type* for $C_0$ and $\mathcal{T}$ if it satisfies the following properties:

- $D_0 \in t$;
- for every concept description $\neg C \in \mathcal{S}$, either $C$ or $\neg C$ belongs to $t$;[5]
- for every concept description $C \sqcap D \in \mathcal{S}$, we have that $C \sqcap D \in t$ iff $C \in t$ and $D \in t$;
- for every concept description $C \sqcup D \in \mathcal{S}$, we have that $C \sqcup D \in t$ iff $C \in t$ or $D \in t$.

Given a model $\mathcal{I}$ of $\mathcal{T}$ and an individual $d \in \Delta^{\mathcal{I}}$, the *type of* $d$ is the set

$$t_{\mathcal{I}}(d) := \{C \in \mathcal{S} \mid d \in C^{\mathcal{I}}\}.$$

It is easy to show that the type of an individual in a model of $\mathcal{T}$ really satisfies the conditions stated in the definition of a type.

Intuitively, these conditions take care of the TBox and of the semantics of the Boolean operation. However, we must also take the successor constraints into account. Given a type $t$, the (possibly negated) successor constraints in $t$ induce a QFBAPA formula $\phi_t$ in the obvious way.[6] Obviously, if $t = t_{\mathcal{I}}(d)$ for an individual in a model of $\mathcal{T}$, then the corresponding QFBAPA formula $\phi_t$ has a solution in which the universal set $\mathcal{U}$ consists of all the role successors of $d$, and the other set

[5]Note the exclusive or, i.e., it is not possible that a concept description *and* its negation is contained in a type.

[6]This is just like the QFBAPA formula $\phi$ obtained from a Boolean valuation in our PSpace algorithm in the previous section.

variables are assigned sets according to the interpretations of roles and concept descriptions in the model. In order to do type elimination, however, we also need to know which are the non-empty Venn regions in this solution. Again, it is sufficient to look at solutions for which only a polynomial number of Venn regions are non-empty.

To be more precise, given a type $t$, we consider the corresponding QFBAPA formula $\phi_t$, and conjoin to this formula the set constraint $X_{r_1} \cup \ldots \cup X_{r_n} = \mathcal{U}$, where $N_R = \{r_1, \ldots, r_n\}$. For the resulting formula $\phi'_t$, we compute the number $N_t$ that bounds the number of Venn regions that need to be non-empty in a solution of $\phi'_t$ (see Lemma 6).

**Definition 9** An *augmented type* $(t, V)$ for $C_0$ and $\mathcal{T}$ consists of a type $t$ for $C_0$ and $\mathcal{T}$ together with a set of Venn region $V$ such that $|V| \leq N_t$ and the formula $\phi'_t$ has a solution in which exactly the Venn regions in $V$ are non-empty.

The existence of a solution of $\phi'_t$ in which exactly the Venn regions in $V$ are non-empty can obviously be checked (within NP) by adding to $\phi'_t$ conjuncts that state non-emptiness of the Venn regions in $V$ and the fact that the union of these Venn regions is the universal set (see the description of the PSpace algorithm in the proof of Theorem 7). Another easy to show observation is that there are only exponentially many augmented types.

**Lemma 10** *The set of augmented types for $C_0$ and $\mathcal{T}$ contains at most exponentially many elements in the size of $C_0$ and $D_0$ and it can be computed in exponential time.*

*Proof.* Since the cardinality of the set $\mathcal{S}$ is polynomial in the size of $C_0$ and $D_0$, there are only exponentially many subsets of $\mathcal{S}$, and for each of these subsets it can be tested in polynomial time whether it satisfies the conditions required for a type. Now, for each type $t$ we can compute the number $N_t$ in polynomial time and the value of this number is polynomial in the size of $\phi'_t$, and thus of the size of $C_0$ and $D_0$. There are exponentially many Venn regions, and thus the set of all sets of Venn regions has double-exponential cardinality. However, there are only exponentially many such sets of cardinality $\leq N_t$, and we can generate all of them in exponential time. Given such a set $V$, we have already argued above why we can test (in NP, and thus in exponential time) whether $(t, V)$ satisfies the conditions required for an augmented type. □

Basically, type elimination starts with the set of all augmented types, and then successively eliminates augmented types whose Venn regions are not realized by the currently available augmented types. To make this more precise, assume that $\mathcal{A}$ is a set of augmented types and that $v$ is a Venn region. The Venn region $v$

yields a concept description $C_v$ (see the description of the PSpace algorithm in the proof of Theorem 7), and it is easy to see that $C_v$ is actually a conjunction of elements of $\mathcal{S}$ (modulo removal of double negation). We say that $v$ is *realized by* $\mathcal{A}$ if there is an augmented type $(t, V) \in \mathcal{A}$ such that every conjunct of $C_v$ is an element of $t$.

**Theorem 11** *Satisfiability of $\mathcal{ALCSCC}$ concept descriptions w.r.t. a TBox is ExpTime-complete.*

*Proof.* Given an $\mathcal{ALCSCC}$ concept description $C_0$ and a TBox $\mathcal{T} = \{\top \sqsubseteq D_0\}$, the type elimination algorithm for deciding satisfiability of $C_0$ w.r.t. $\mathcal{T}$ proceeds as follows:

1. Compute the set $\mathcal{S}$ consisting of all subdescriptions of $C_0$ and $D_0$ as well as the negations of these subdescriptions, and continue with the next step.
2. Based on $\mathcal{S}$, compute the set $\mathcal{A}$ of all augmented types for $C_0$ and $\mathcal{T}$, and continue with the next step.
3. If the current set $\mathcal{A}$ of augmented types is empty, then the algorithm *fails*. Otherwise, check whether $\mathcal{A}$ contains an element $(t, V)$ such that not all the Venn regions in $V$ are realized by $\mathcal{A}$. If there is no such element $(t, V)$ in $\mathcal{A}$, then continue with the next step. Otherwise, let $(t, V)$ be such an element, and set $\mathcal{A} := \mathcal{A} \setminus \{(t, V)\}$. Continue with this step, but now using the new current set of augmented types.
4. If $\mathcal{A}$ contains an augmented type $(t, V)$ such that $C_0 \in t$, then the algorithm *succeeds*. Otherwise, the algorithm *fails*.

This algorithm indeed runs *in exponential time* since

- Step 1 can obviously be performed in polynomial time;
- according to Lemma 10, Step 2 can be performed in exponential time;
- Step 3 can be iterated only an exponentially number of times since each time one augmented type is removed, and there are only exponentially many to start with. Every single execution of Step 3 takes exponential time since at most exponentially many augmented types and Venn regions need to be considered when testing whether every Venn region occurring in an augmented type of $\mathcal{A}$ is realized in $\mathcal{A}$;
- in Step 4, at most exponentially many augmented types need to be checked as to whether their first component contains $C_0$.

Regarding *soundness* (i.e., if the algorithm succeeds, then $C_0$ is indeed satisfiable w.r.t. $\mathcal{T}$), we show how the set of augmented types $\mathcal{A}$ computed by a successful run of the algorithm can be used to construct a model $\mathcal{I}$ of $C_0$ w.r.t. $\mathcal{T}$. This model contains, for every augmented type $(t, V) \in \mathcal{A}$ countably infinitely many copies, i.e.,

$$\Delta^{\mathcal{I}} := \{(t, V)^i \mid (t, V) \in \mathcal{A} \text{ and } i \geq 0\}.$$

The interpretation of the concept names $A$ is based on the occurrence of these names in the first component of an augmented type, i.e.,

$$A^{\mathcal{I}} := \{(t, V)^i \in \Delta^{\mathcal{I}} \mid A \in t\}.$$

Defining the interpretation of the role names is a bit more tricky. Obviously, it is sufficient to define, for each role name $r \in N_R$ and each $d \in \Delta^{\mathcal{I}}$, the set $r^{\mathcal{I}}(d)$. Thus, consider an element $(t, V)^i \in \Delta^{\mathcal{I}}$. Since $(t, V)$ is an augmented type in $\mathcal{A}$, the formula $\phi'_t$ has a solution $\sigma$ in which exactly the Venn regions in $V$ are non-empty. In addition, each Venn region $v \in V$ is realized by an augmented type $(t^v, V^v) \in \mathcal{A}$. Assume that the solution $\sigma$ assigns the finite set $\{d_1, \ldots, d_m\}$ to the set term $\mathcal{U}$. We consider a injective mapping $\pi$ of $\{d_1, \ldots, d_m\}$ into $\Delta^{\mathcal{I}}$ such that the following holds for each element $d_j$ of $\{d_1, \ldots, d_m\}$: if $d_j$ belongs to the Venn region $v \in V$, then $\pi(d_j) = (t^v, V^v)^\ell$ for some $\ell \geq 0$. We now define

$$r^{\mathcal{I}}((t, V)^i) := \{\pi(d_j) \mid d_j \in \sigma(X_r)\}.$$

Soundness of our algorithm is now an easy consequence of the following claim:

**Claim:** *For all $C \in \mathcal{S}$, $(t, V) \in \mathcal{A}$, and $i \geq 0$ we have $C \in t$ iff $(t, V)^i \in C^{\mathcal{I}}$.*

We prove the claim by induction on the size of $C$:

- If $C = A$ for a concept name $A$, then the claim immediately follows from the definition of $A^{\mathcal{I}}$.
- If $C = \neg D$, then we know by induction that $D \in t$ iff $(t, V)^i \in D^{\mathcal{I}}$. Thus we have $C \in t$ iff $D \notin t$ iff $(t, V)^i \notin D^{\mathcal{I}}$ iff $(t, V)^i \in C^{\mathcal{I}}$, where the first equivalence follows from the definition of types, the second by induction, and the third by the semantics of negation.
- If $C = D_1 \sqcap D_2$, then we similarly have $C \in t$ iff $D_1 \in t$ and $D_2 \in t$ iff $(t, V)^i \in D_1$ and $(t, V)^i \in D_2$ iff $(t, V)^i \in C^{\mathcal{I}}$.
- The case $C = D_1 \sqcup D_2$ can be handled analogously.
- Now assume that $C = \mathit{succ}(c)$ for a set or cardinality constraint $c$.
  - If $C \in t$, then this constraint is part of the QFBAPA formula $\phi'_t$ obtained from $t$, and thus satisfied by the solution $\sigma$ of $\phi'_t$ used to

define the role successors of $(t, V)^i$. According to this definition, there is a 1–1 correspondence between the elements of $\sigma(\mathcal{U})$ and the role successors of $(t, V)^i$. This bijection $\pi$ also respects the assignment of subsets of $\sigma(\mathcal{U})$ to set variables of the form $X_r$ (for $r \in N_R$) and $X_D$ (for concept descriptions $D$) occurring in $\phi'_t$, i.e.,

$(*)$ $d_j \in \sigma(X_r)$ iff $\pi(d_j) \in r^{\mathcal{I}}((t, V)^i)$ and $d_j \in \sigma(X_D)$ iff $\pi(d_j) \in D^{\mathcal{I}}$.

Once $(*)$ is shown it is clear that $(t, V)^i \in \mathit{succ}(c)^{\mathcal{I}} = C^{\mathcal{I}}$. In fact, the translation $\phi_c$ of $c$, where $r$ is replaced by $X_r$ and $D$ by $X_D$, is a conjunct in $\phi'_t$ and thus $\sigma$ satisfies $\phi_c$. Now $(*)$ shows that the mapping $\mathcal{I}^d$ for $d = (t, V)^i$ coincides with $\sigma$ (modulo the replacement of $r$ by $X_r$ and $D$ by $X_D$), and thus the fact that $\sigma$ satisfies $\phi_c$ implies that $\mathcal{I}^d$ satisfies $c$.

For role names $r$, property $(*)$ is immediate by the definition of $r^{\mathcal{I}}((t, V)^i)$. Now consider a concept description $D$ such that $X_D$ occurs in $\phi'_t$. Then $D$ occurs in $c$, and is thus smaller than $C$, which means that we can apply induction to it. If $d_j \in \sigma(X_D)$, then the Venn region $v$ to which $d_j$ belongs contains $X_D$ positively. Consequently, $C_v$ contains $D$ as a conjunct, and the augmented type $(t^v, V^v)$ realizing $v$ satisfies $D \in t_v$. By induction, we obtain $\pi(d_j) = (t^v, V^v)^\ell \in D^{\mathcal{I}}$. Conversely, assume that $\pi(d_j) = (t^v, V^v)^\ell \in D^{\mathcal{I}}$, where $v$ is the Venn region to which $d_j$ belongs w.r.t. $\sigma$. By induction, we obtain $D \in t_v$, and thus the Venn region $v$ contains $X_D$ positively. Since $d_j$ belongs to this Venn region, we obtain $d_j \in \sigma(X_D)$.

– The case where $C \notin t$ can be treated similarly. In fact, in this case the constraint $\neg c$ is part of the QFBAPA formula $\phi'_t$ obtained from $t$, and we can employ the same argument as above, just using $\neg c$ instead of $c$.

This completes the last case of our induction proof, and thus finishes the proof of the claim.

The claim can now be used to show that $\mathcal{I}$ is indeed a model of $C_0$ w.r.t. $\mathcal{T}$. Firstly, since every augmented type $(t, V) \in \mathcal{A}$ satisfies $D_0 \in t$ by Definition 8, the claim yields $(t, V)^i \in D_0^{\mathcal{I}}$ for all $i \geq 0$, and thus every element of $\Delta^{\mathcal{I}}$ satisfies the GCI $\top \sqsubseteq D_0$. Consequently, $\mathcal{I}$ is indeed a model of $\mathcal{T}$. Secondly, since the algorithm has terminated successfully, $\mathcal{A}$ contains an augmented type $(t, V)$ such that $C_0 \in t$. This implies $(t, V)^i \in C_0^{\mathcal{I}}$ for all $i \geq 0$, and thus $\mathcal{I}$ indeed contains a witness (actually, infinitely many) for the satisfaction of $C_0$ in $\mathcal{I}$. This completes the proof of soundness of our algorithm.

To show *completeness* (i.e., if $C_0$ is satisfiable w.r.t. $\mathcal{T}$, then the algorithm succeeds), assume that $\mathcal{I}$ is a model of $C_0$ with witness $d_0$. Consider the set of all

types of elements of $\mathcal{I}$, i.e.,

$$t(\mathcal{I}) := \{t_{\mathcal{I}}(d) \mid d \in \Delta^{\mathcal{I}}\}.$$

Since $\mathcal{I}$ is a model of $\mathcal{T}$, the set $t(\mathcal{I})$ is indeed a set of types according to Definition 8. In addition, we have $C_0 \in t_{\mathcal{I}}(d_0)$ since $d_0$ is a witness for the satisfaction of $C_0$. Let us now extend the types in $t(\mathcal{I})$ by adding appropriate Venn regions as second components. Consider $t := t_{\mathcal{I}}(d)$ for an element $d \in \Delta^{\mathcal{I}}$. Then the QFBAPA formula $\phi'_t$ corresponding to $t$ has a solution $\sigma$ in which the universal set $\mathcal{U}$ consists of all the role successors of $d$, and the other set variables are assigned sets according to the interpretations of roles and concept descriptions in the model $\mathcal{I}$. Let $\{d_1, \ldots, d_m\} = \sigma(\mathcal{U})$ be the set of all role successors of $d$, and $v_i$ the Venn region to which $d_i$ belongs w.r.t. $\sigma$. By Lemma 6, there is a solution $\sigma'$ of $\phi'_t$ such that the set $V$ of non-empty Venn regions w.r.t. $\sigma'$ has cardinality $\leq N_t$ and each of these non-empty Venn regions in $V$ is one of the Venn regions $v_i$, i.e., $V \subseteq \{v_1 \ldots, v_m\}$. By construction, $(t, V)$ is an augmented type. Let $\mathcal{A}(\mathcal{I})$ denote the set of augmented types obtained by extending the types in $t(\mathcal{I})$ in this way.

We claim that the Venn regions occurring in some augmented type in $\mathcal{A}(\mathcal{I})$ are realized by $\mathcal{A}(\mathcal{I})$. Thus, let $(t, V)$ be an augmented type constructed from a type $t = t_{\mathcal{I}}(d)$ as described above, and let $v \in V$ be a Venn region occurring in this augmented type. Then there is a role successor $d_i$ of $d$ such that $d_i$ belongs to the Venn region $v = v_i$ w.r.t. the solution $\sigma$ of $\phi'_t$ used in the construction. We know that $d_i \in C^{\mathcal{I}}_{v_i}$, and thus every conjunct of $C_v = C_{v_i}$ is an element of $t_{\mathcal{I}}(d_i)$. Since $\mathcal{A}(\mathcal{I})$ contains an augmented type with first component $t_{\mathcal{I}}(d_i)$, this shows that $v$ is realized by $\mathcal{A}(\mathcal{I})$.

Using the fact that the Venn regions occurring in some augmented type in $\mathcal{A}(\mathcal{I})$ are realized by $\mathcal{A}(\mathcal{I})$, it is easy to show that no element of $\mathcal{A}(\mathcal{I})$ can be removed during type elimination, i.e., during the whole run of the algorithm we have $\mathcal{A}(\mathcal{I}) \subseteq \mathcal{A}$. Since $\mathcal{A}(\mathcal{I})$ is non-empty (due to the fact that $\Delta^{\mathcal{I}} \neq \emptyset$) and contains an augmented type with first component $t_{\mathcal{I}}(d_0)$, this shows that the algorithm cannot fail. This completes the proof of completeness, and thus of the theorem. □

# 7 Related work and future work

The work most closely related to ours is the one by Ohlbach and Koehler [14] and by Demri and Lugiez [5].

Demri and Lugiez introduce the modal logic EML, which allows for arithmetic constraints on the cardinality of successors w.r.t. the transition relations $\mathtt{R}$ and for automata-based constraints of the form $\mathcal{A}^{\mathtt{R}}(\phi_1, \ldots, \phi_n)$. Here, $\mathcal{A}$ is a finite automaton on finite words, for which there is a 1–1-relationship between the

formulae $\phi_1, \ldots, \phi_n$ and the alphabet $\Sigma = \{a_1, \ldots, a_n\}$ over which the words are built. A world (i.e., element of a Kripke structure) satisfies the constraint $\mathcal{A}^{\mathtt{R}}(\phi_1, \ldots, \phi_n)$ if the ordered collection of $\mathtt{R}$-successors of this world satisfies a pattern accepted by $\mathcal{A}$. For example, if $n = 2$ and $\mathcal{A}$ accepts the regular language $a_1 a_2^* a_1$, the world needs to have finitely many successors such that the first and the last one satisfy $\phi_1$ and the other successors satisfy $\phi_2$.

Demri and Lugiez show that the satisfiability problem is PSpace-complete for EML. Using the well-known connection between DLs and modal logics [17], it is easy to see that Demri and Lugiez's logic EML without the automata-based constraints is equivalent to the logic introduced in the present paper. Thus, their "in PSpace" result implies our Theorem 7; however their proof of this result is considerably more complex than ours due to the presence of automata-based constraints. But even disregarding the treatment of these constraints, their proof is quite different from ours. Since this is not usually considered for modal logics, Demri and Lugiez do *not* show a result corresponding to our ExpTime result for satisfiability w.r.t. a TBox.

Like ours, the work by Ohlbach and Koehler also allows for Boolean set terms and arithmetic constraints on the cardinality of role successors. On the one hand, this work is more general than ours in that the authors allow also for bridging functions other than cardinality from successors sets into the arithmetic domain. Actually, while the authors of [14] use the cardinality function in most of their examples, the formal problem specification (Definition 4 in [14]) only requires the bridging functions to satisfy an additivity axiom (Definition 3 in [14]), which in the case of cardinality says:

$$\text{If } x \cap y = \emptyset \text{ then } |x \cup y| = |x| + |y|.$$

It is not clear whether reasoning is done w.r.t. all possible bridging functions satisfying the additivity axiom or w.r.t. specific bridging functions such as cardinality.

On the other hand, the set expressions in [14] can only contain roles and not complex concept descriptions. However, a combination of value restrictions on subroles and cardinality constraints on these subroles can simulate this expressiveness. For example, as pointed out in [14], a qualified number restriction such as $\geqslant n\, r.C$ can be expressed as $succ(r' \subseteq C) \sqcap succ(r' \subseteq r) \sqcap succ(|r'| \geq n)$, where $r'$ is a newly introduced role name.[7] Similarly, $\leqslant n\, r.C$ can be expressed as $succ(r' \subseteq C) \sqcap succ(r \cap r'^c \subseteq \neg C) \sqcap succ(r' \subseteq r) \sqcap succ(|r'| \leq n)$. More generally, one can replace the concept description $C$ within a successor constraint by the new role name $r'$ if one conjoins $r' \subseteq C$ and $r'^c \subseteq \neg C$ to this constraint.

The major difference to our work is, however, that Ohlbach and Koehler [14] give

[7] Note that [14] actually uses a different syntax for cardinality restrictions on role successors. To avoid having to introduce another syntax, we have translated this into our syntax. The constraint $succ(r' \subseteq C)$ expresses the value restriction $\forall r'.C$.

only decidability results and no complexity results. Due to the fact that they consider all Venn regions and also resolve Boolean reasoning on the Description Logic side using disjunctive normal form, the complexity of their decision procedures is considerably higher than the upper bounds we show. In addition, they do not consider GCIs in their work. Even without GCIs, the complexity of the unoptimized procedure in [14] is probably non-deterministic-exponential since an NP procedure solving the arithmetic constraints is applied to a potentially exponentially large constraint system.

The emphasis of the current paper was on showing worst-case optimal complexity results, and thus the algorithms as described here cannot directly be used for implementation purposes. To make the PSpace algorithm more practical, guessing would need to be replaced by SAT solving. Such an algorithm would need to combine (similarly to SMT solvers) an efficient SAT solver with a solver for QF-BAPA and with a recursive application of itself. Type elimination is exponential also in the best case since it first computes an exponential number of (augmented) types and only then starts the elimination process. Instead, one could use an algorithm similar to the practically more efficient version of the PSpace algorithm just sketched. However, due to the presence of GCIs, the recursion depth of recursive calls is no longer bounded. Thus, one would need to ensure termination by an appropriate blocking strategy, similar to what tableau-based algorithms use. One could also try to design tablau-based satisfiability algorithms, but then needs to be very careful to avoid the problems caused by the "naive idea" sketched below Example 5 when backtracking.

**Acknowledgment.** The author thanks Viktor Kuncak for helpful discussions regarding the proof of Lemma 6.

# References

[1] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[2] Franz Baader and Ulrike Sattler. Expressive number restrictions in description logics. *J. of Logic and Computation*, 9(3):319–350, 1999.

[3] Alexander Borgida, Ronald J. Brachman, Deborah L. McGuinness, and Lori Alperin Resnick. CLASSIC: A structural data model for objects. In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 59–67, 1989.

[4] Maarten de Rijke. A note on graded modal logic. *Studia Logica*, 64(2):271–283, 2000.

[5] Stéphane Demri and Denis Lugiez. Complexity of modal logics with presburger constraints. *J. Applied Logic*, 8(3):233–252, 2010.

[6] Jocelyne Faddoul and Volker Haarslev. Algebraic tableau reasoning for the description logic SHOQ. *J. Applied Logic*, 8(4):334–355, 2010.

[7] Michael R. Garey and David S. Johnson. *Computers and Intractability — A guide to NP-completeness*. W. H. Freeman and Company, San Francisco (CA, USA), 1979.

[8] Volker Haarslev, Roberto Sebastiani, and Michele Vescovi. Automated reasoning in $\mathcal{ALCQ}$ via SMT. In *Proc. of the 23rd Int. Conf. on Automated Deduction (CADE 2011)*, volume 6803 of *Lecture Notes in Computer Science*, pages 283–298. Springer-Verlag, 2011.

[9] Volker Haarslev, Martina Timmann, and Ralf Möller. Combining tableaux and algebraic methods for reasoning with qualified number restrictions. In *Proc. of the 2001 Description Logic Workshop (DL 2001)*, volume 49 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2001.

[10] Robert Hoehndorf, Paul N. Schofield, and Georgios V. Gkoutos. The role of ontologies in biological and biomedical research: A functional perspective. *Brief. Bioinform.*, 16(6):1069–1080, 2015.

[11] Bernhard Hollunder and Franz Baader. Qualifying number restrictions in concept languages. In *Proc. of the 2nd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'91)*, pages 335–346, 1991.

[12] Bernhard Hollunder, Werner Nutt, and Manfred Schmidt-Schauß. Subsumption algorithms for concept description languages. In *Proc. of the 9th Eur. Conf. on Artificial Intelligence (ECAI'90)*, pages 348–353, London (United Kingdom), 1990. Pitman.

[13] Viktor Kuncak and Martin C. Rinard. Towards efficient satisfiability checking for Boolean algebra with Presburger arithmetic. In Frank Pfenning, editor, *Proc. of the 21st Int. Conf. on Automated Deduction (CADE-07)*, volume 4603 of *Lecture Notes in Computer Science*, pages 215–230. Springer, 2007.

[14] Hans Jürgen Ohlbach and Jana Koehler. Modal logics, description logics and arithmetic reasoning. *Artificial Intelligence*, 109(1–2):1–31, 1999.

[15] V. R. Pratt. Models of program logic. In *Proc. of the 20th Annual Symp. on the Foundations of Computer Science (FOCS'79)*, pages 115–122, 1979.

[16] Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. Type-elimination-based reasoning for the description logic $\mathcal{SHIQ}\mathrm{b}_s$ using decision diagrams and disjunctive datalog. *Logical Methods in Computer Science*, 8(1), 2012.

[17] Klaus Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 466–471, 1991.

[18] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[19] Stephan Tobies. A PSPACE algorithm for graded modal logic. In Harald Ganzinger, editor, *Proc. of the 16th Int. Conf. on Automated Deduction (CADE'99)*, volume 1632 of *Lecture Notes in Artificial Intelligence*, pages 52–66. Springer-Verlag, 1999.