Concrete Domains Meet Expressive Cardinality Restrictions in Description Logics

Franz Baader[1,2], Stefan Borgwardt[1], Filippo De Bortoli[1,2] ✉, and Patrick Koopmann[3]

[1] TU Dresden, Institute of Theoretical Computer Science
Dresden, Germany
[2] Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI)
Dresden/Leipzig, Germany
[3] Vrije Universiteit Amsterdam, Department of Computer Science
Amsterdam, The Netherlands

Abstract. Standard Description Logics (DLs) can encode quantitative aspects of an application domain through either *number restrictions*, which constrain the number of individuals that are in a certain relationship with an individual, or *concrete domains*, which can be used to assign concrete values to individuals using so-called features. These two mechanisms have been extended towards very expressive DLs, for which reasoning nevertheless remains decidable. Number restrictions have been generalized to more powerful comparisons of sets of role successors in $\mathcal{ALCSCC}$, while the comparison of feature values of different individuals in $\mathcal{ALC}(\mathfrak{D})$ has been studied in the context of ω-admissible concrete domains $\mathfrak{D}$. In this paper, we combine both formalisms and investigate the complexity of reasoning in the thus obtained DL $\mathcal{ALCOSCC}(\mathfrak{D})$, which additionally includes the ability to refer to specific individuals by name. We show that, in spite of its high expressivity, the consistency problem for this DL is ExpTime-complete, assuming that the constraint satisfaction problem of $\mathfrak{D}$ is also decidable in exponential time. It is thus not higher than the complexity of the basic DL $\mathcal{ALC}$. At the same time, we show that many natural extensions to this DL, including a tighter integration of the concrete domain and number restrictions, lead to undecidability.

Keywords: Description Logics · Automated Deduction · Concrete Domains · Cardinality Constraints

1 Introduction

Description logics (DLs) [6,9] are a well-investigated family of logic-based knowledge representation languages, which can be used to formalize the terminological knowledge of an application domain in a machine-processable way. For instance, the popular Web Ontology Language OWL[4] is based on an expressive DL and

[4] https://www.w3.org/TR/owl2-overview/

large medical ontologies such as SNOMED CT[5] and Galen[6] have been developed using appropriate DLs. A key feature of DLs is the ability to construct descriptions of complex concepts (i.e., sets of individuals sharing certain properties) using concept names (unary predicates) and role names (binary predicates). For example, the concept of a parent can be described as $\mathsf{Human} \sqcap \exists \mathsf{child}.\mathsf{Human}$. Knowledge about the relationship between concepts can then be expressed using concept inclusions (CIs), such as $\mathsf{Human} \sqcap \exists \mathsf{child}.\mathsf{Human} \sqsubseteq \exists \mathsf{eligible}.\mathsf{TaxBreak}$, which says that parents are eligible for a tax break.

Such purely qualitative statements are not always sufficient to express quantitative information (e.g. the number of children required for a tax break) that is relevant for an application domain. *Qualified number restrictions* that constrain the number of role successors belonging to a certain concept by a fixed natural number can be employed in DLs to express such quantitative information; e.g., $\mathsf{Human} \sqcap (\geq 3\, \mathsf{child}.\mathsf{Human}) \sqsubseteq \exists \mathsf{eligible}.\mathsf{TaxBreak}$ says that a tax break is available if one has at least three children. *Concrete domain restrictions* can represent a different type of quantitative information, where concrete objects such as numbers or strings can be assigned to individuals using partial functions (*features*). For example, a tax break might only be available if the annual salary is not too high. The CI $\mathsf{Human} \sqcap (\geq 3\, \mathsf{child}.\mathsf{Human}) \sqcap \exists \mathsf{salary}.<_{100{,}000} \sqsubseteq \exists \mathsf{eligible}.\mathsf{TaxBreak}$ specifies at least three children and an annual salary of less than 100,000 € as eligibility criteria for a tax break.

Both (qualified) number restrictions [18,17] and concrete domain restrictions [8] have been introduced early on in DL research, but it turned out that they create considerable algorithmic challenges. For $\mathcal{ALCQ}$, the extension of the basic DL $\mathcal{ALC}$ with qualified number restrictions, it was open for a decade whether the increase in expressivity also increases the complexity of reasoning if numbers in number restrictions are assumed to be represented in binary, until Tobies [31,30] was able to show that it stays the same (PSpace without and ExpTime with CIs). It also turned out that the unrestricted use of transitive roles within number restrictions can cause undecidability [20]. In [3], it was shown that reasoning in $\mathcal{ALCSCC}$, which extends $\mathcal{ALCQ}$ with very expressive counting constraints on role successors expressed in the logic QFBAPA [22], still has the same complexity as in $\mathcal{ALC}$ and $\mathcal{ALCQ}$. In this logic, one can, e.g., describe humans that have exactly as many cars as children as $\mathsf{Human} \sqcap \mathsf{succ}(|\mathsf{own} \cap \mathsf{Car}| = |\mathsf{child} \cap \mathsf{Human}|)$, without having to specify the exact numbers of cars and children. Such statements cannot even be expressed in full first-order logic [7].

The decidability result for $\mathcal{ALC}(\mathfrak{D})$, i.e., $\mathcal{ALC}$ extended with an admissible concrete domain $\mathfrak{D}$, in [8] did not take CIs into account. In the presence of CIs, integrating even rather simple concrete domains into the DL $\mathcal{ALC}$ may cause undecidability [26,10]. In [27], it was proved that integrating a so-called ω-admissible concrete domain into $\mathcal{ALC}$ leaves reasoning decidable also in the presence of CIs. That paper gives two examples of such concrete domains (Allen's interval algebra [1] and RCC8 [29]). Using well-known notions and results from

[5] https://www.snomed.org/

[6] https://bioportal.bioontology.org/ontologies/GALEN

model theory, additional ω-admissible concrete domains were exhibited in [10,11], for example the rational numbers with comparisons $\mathfrak{Q} := (\mathbb{Q}, <, =, >)$. Decidability results for $\mathcal{ALC}(\mathfrak{D})$ in the presence of CIs for concrete domains $\mathfrak{D}$ that are not ω-admissible can be found in [14,23,15]. A simpler, but considerably more restrictive way of achieving decidability is to use unary concrete domains [19].

In this paper, we study $\mathcal{ALCOSCC}(\mathfrak{D})$, a combination of the DLs $\mathcal{ALCSCC}$ and $\mathcal{ALC}(\mathfrak{D})$ with ω-admissible concrete domains $\mathfrak{D}$ as well as nominals ($\mathcal{O}$). However, our logic goes beyond a pure combination of number restrictions and concrete domains by additionally allowing them to interact. For a numerical concrete domain, it seems natural to use the values of concrete features directly in the QFBAPA constraints, e.g. to describe people that own more books than their age. We show, however, that this unrestricted combination easily leads to undecidability. Instead, we use concrete domain constraints to define roles, which can then be employed within QFBAPA constraints. For example, the *concrete role* $(\mathsf{salary} < \mathtt{next}\,\mathsf{salary})$ connects an individual to all individuals that have a higher salary. One can use this to describe all persons that have a lower salary than at least half of their children with $\mathsf{succ}(|\mathsf{child} \cap (\mathsf{salary} < \mathtt{next}\,\mathsf{salary})| > |\mathsf{child} \cap (\mathsf{salary} \geq \mathtt{next}\,\mathsf{salary})|)$. However, we show that the unrestricted use of such concrete roles also leads to undecidability. Hence, we additionally restrict them to pairs of individuals that are already connected by a role name.

Our main result is that the complexity of reasoning in $\mathcal{ALCOSCC}(\mathfrak{D})$ stays in ExpTime if the complexity of reasoning in $\mathfrak{D}$ is in ExpTime. There are few results in the literature that determine the exact complexity of reasoning in DLs with concrete domains [25,23,15,13]. Only [25] and [13] consider ω-admissible concrete domains, and the ExpTime-completeness result in the former is restricted to a specific temporal concrete domain. Our paper extends the results of the latter from $\mathcal{ALC}(\mathfrak{D})$ to $\mathcal{ALCOSCC}(\mathfrak{D})$ and is generic since it holds for all ω-admissible concrete domains with a decision problem in ExpTime. Finally, apart from the aforementioned undecidability results, we show that adding transitive roles also makes reasoning undecidable, even under strong syntactic restrictions. All proof details can be found in [5].

2 Preliminaries

Concrete domains. We adopt the term *concrete domain* to refer to a relational structure $\mathfrak{D} = (D, P_1^D, P_2^D, \dots)$ over a non-empty, countable relational signature $\{P_1, P_2, \dots\}$, where D is a non-empty set, and each predicate P_i has an associated arity $k_i \in \mathbb{N}$ and is interpreted by a relation $P_i^D \subseteq D^{k_i}$. An example is the structure $\mathfrak{Q} := (\mathbb{Q}, <, =, >)$ over the rational numbers $\mathbb{Q}$ with standard binary order and equality relations. Given a countably infinite set V of variables, a *constraint system* over V is a set $\mathfrak{C}$ of *constraints* $P(v_1, \dots, v_k)$, where $v_1, \dots, v_k \in V$ and P is a k-ary predicate of $\mathfrak{D}$. We denote by $V(\mathfrak{C})$ the set of variables that occur in $\mathfrak{C}$. The constraint system $\mathfrak{C}$ is *satisfiable* if there is a mapping $h\colon V(\mathfrak{C}) \to D$, called *solution* of $\mathfrak{C}$, such that $P(v_1, \dots, v_k) \in \mathfrak{C}$ implies $(h(v_1), \dots, h(v_k)) \in P^D$. The *constraint satisfaction problem* for $\mathfrak{D}$, denoted

CSP($\mathfrak{D}$), asks if a given finite constraint system $\mathfrak{C}$ over $\mathfrak{D}$ is satisfiable. The CSP of $\mathfrak{Q}$ is decidable in polynomial time, by reduction to <-cycle detection: for example, the clique $x_1 < x_2,\ x_2 < x_3,\ x_3 < x_1$ is unsatisfiable over $\mathfrak{Q}$.

To ensure that reasoning in DLs with concrete domain restrictions remains decidable, we impose further properties on $\mathfrak{D}$ regarding its predicates and the compositionality of its CSP for finite and countable constraint systems. We say that $\mathfrak{D}$ is a *patchwork* if it satisfies the following conditions:[7]

JEPD if $k \geq 1$ and $\mathfrak{D}$ has k-ary predicates, then these predicates partition D^k;
JD there is a quantifier-free, equality-free first-order formula $\phi_=(x, y)$ over the signature of $\mathfrak{D}$ that defines equality between two elements of $\mathfrak{D}$;
AP if $\mathfrak{B}$, $\mathfrak{C}$ are constraint systems and $P(v_1, \ldots, v_k) \in \mathfrak{B}$ iff $P(v_1, \ldots, v_k) \in \mathfrak{C}$ holds for all $v_1, \ldots, v_k \in V(\mathfrak{B}) \cap V(\mathfrak{C})$ and all k-ary predicates P over $\mathfrak{D}$, then $\mathfrak{B}$ and $\mathfrak{C}$ are satisfiable iff $\mathfrak{B} \cup \mathfrak{C}$ is satisfiable.

If $\mathfrak{D}$ is a patchwork, we call a constraint system $\mathfrak{C}$ *complete* if, for all $k \in \mathbb{N}$ for which $\mathfrak{D}$ has k-ary predicates and all $v_1, \ldots, v_k \in V(\mathfrak{C})$, there is exactly one k-ary predicate P over $\mathfrak{D}$ such that $P(v_1, \ldots, v_k) \in \mathfrak{C}$. The concrete domain $\mathfrak{D}$ is *homomorphism ω-compact* if every countable constraint system $\mathfrak{C}$ over $\mathfrak{D}$ is satisfiable whenever all its finite subsystems $\mathfrak{C}' \subseteq \mathfrak{C}$ are satisfiable. We introduce ExpTime-ω-admissible concrete domains, which differ from ω-admissible ones as defined in [27,11] by a stronger requirement on the decidability of CSP($\mathfrak{D}$).

Definition 1. *A concrete domain $\mathfrak{D}$ is* ExpTime-ω-admissible *if it has a finite signature, it is a patchwork, it is homomorphism ω-compact and its CSP is in* ExpTime.

The finiteness of the signature of $\mathfrak{D}$ is necessary to ensure decidability. Without this assumption, one can find instances of $\mathfrak{D}$ that satisfy all the other conditions of Definition 1 such that reasoning in $\mathcal{ALC}(\mathfrak{D})$ is undecidable. One such example is given by the concrete domain $(\mathbb{Z}, \{+_m \mid m \in \mathbb{Z}\})$ where $+_m$ relates those integers whose difference is equal to m [11]. The conditions of Definition 1 are satisfied by Allen's interval algebra, RCC8 and $\mathfrak{Q}$ [27,11].

The logic QFBAPA. Set terms are built using the operations intersection $\cap$, union $\cup$ and complement ^{c} from *set variables* and the constants $\emptyset$ and $\mathcal{U}$. Set terms s, t are then used in *inclusion-* and *equality constraints* $s \subseteq t$, $s = t$. *Presburger Arithmetic (PA) expressions* ℓ, ℓ' of the form $n_0 + n_1 t_1 + \cdots + n_k t_k$, where $n_i \in \mathbb{N}$ and each t_i is the cardinality $|s_i|$ of a set term s_i or an integer variable, are used to form *numerical constraints* $\ell = \ell'$, $\ell < \ell'$ and n div ℓ (n divides ℓ), where $n \in \mathbb{N}$. A *QFBAPA formula* is a Boolean combination of set- and numerical constraints.

A *solution* σ of a QFBAPA formula ϕ 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 ϕ is *satisfied* by σ, which is defined in the standard way. ϕ is *satisfiable* if it has a solution.

[7] Originally [27] used JEPD (jointly exhaustive, pairwise disjoint) and AP (amalgamation property), while JD (jointly diagonal) was later added by [11].

The satisfiability problem for QFBAPA formulae is NP-complete. Membership in NP is proved in [22], using the notion of Venn regions. If ϕ is a QFBAPA formula containing the set variables $X_1, \dots, X_k$, a *Venn region* for ϕ is a set term of the form $X_1^{c_1} \cap \cdots \cap X_k^{c_k}$ where each c_i is either empty or the set complement c. For a Venn region v for ϕ and a set variable X in ϕ, we write $X \in v$ to indicate that X occurs without complement in v, and $X \notin v$ if X^c occurs in v. The following characterization, proved in [22] and strengthened in [3], guarantees the existence of solutions with a polynomial number of non-empty Venn regions for satisfiable QFBAPA formulae, as follows.

Lemma 1 ([3]). *For every QFBAPA formula ϕ, one can compute in polynomial time a number N_ϕ, whose value is polynomial in the size of ϕ, such that for every solution σ of ϕ there exists a solution σ' fulfilling the following conditions:*

- *there are at most N_ϕ Venn regions v for ϕ for which $\sigma'(v) \neq \emptyset$;*
- *if v is a Venn region for ϕ and $\sigma'(v) \neq \emptyset$, then $\sigma(v) \neq \emptyset$.*

3 Syntax and Semantics of $\mathcal{ALCOSCC}(\mathfrak{D})$

We now introduce the classical description logic $\mathcal{ALCO}$, its extension $\mathcal{ALCOSCC}$, and finally our new logic $\mathcal{ALCOSCC}(\mathfrak{D})$.

Given at most countable, disjoint sets $\mathsf{N_C}$, $\mathsf{N_R}$ and $\mathsf{N_I}$ of *concept-*, *role-* and *individual names*, $\mathcal{ALCO}$ *concepts* are built using negation $\neg$ and conjunction $\sqcap$ from concept names $A \in \mathsf{N_C}$, *nominals* $\{a\}$ with $a \in \mathsf{N_I}$ and *existential restrictions* $\exists r.C$ with $r \in \mathsf{N_R}$ and C an $\mathcal{ALCO}$ concept [9]. As usual, we use $C \sqcup D := \neg(\neg C \sqcap \neg D)$ (disjunction) and $\top := A \sqcup \neg A$. An *interpretation* $\mathcal{I}$ consists of a *domain* $\Delta^\mathcal{I} \neq \emptyset$ and a mapping $\cdot^\mathcal{I}$ assigning sets $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ to $A \in \mathsf{N_C}$, relations $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ to $r \in \mathsf{N_R}$ and individuals $a^\mathcal{I} \in \Delta^\mathcal{I}$ to $a \in \mathsf{N_I}$. For $d \in \Delta^\mathcal{I}$, we define $r^\mathcal{I}(d) := \{e \in \Delta^\mathcal{I} \mid (d, e) \in r^\mathcal{I}\}$. We extend $\cdot^\mathcal{I}$ to concepts by $(\neg C)^\mathcal{I} := \Delta^\mathcal{I} \setminus C^\mathcal{I}$, $(C \sqcap D)^\mathcal{I} := C^\mathcal{I} \cap D^\mathcal{I}$, $\{a\}^\mathcal{I} := \{a^\mathcal{I}\}$ and $(\exists r.C)^\mathcal{I} := \{d \in \Delta^\mathcal{I} \mid \exists e \in r^\mathcal{I}(d) \cap C^\mathcal{I}\}$. In this DL, the concept of all individuals that are human and have a child who is not `Sam` can be written as $\mathsf{Human} \sqcap \exists \mathsf{child}.\neg\{\mathtt{Sam}\}$.

$\mathcal{ALCOSCC}$ extends $\mathcal{ALCO}$ concepts with *role successor restrictions* (or succ-*restrictions*) $\mathsf{succ}(\mathtt{con})$, where `con` is a set- or numerical constraint with role names and concepts as set variables and no integer variables, e.g. $r \subseteq C$ [3]. This DL requires interpretations $\mathcal{I}$ to be *finitely branching*, i.e. such that the set of all role successors $\mathsf{ars}^\mathcal{I}(d) := \bigcup_{r \in \mathsf{N_R}} r^\mathcal{I}(d)$ is finite, for all $d \in \Delta^\mathcal{I}$. Then, each $d \in \Delta^\mathcal{I}$ induces a QFBAPA assignment σ_d, where $\sigma_d(\mathcal{U}) := \mathsf{ars}^\mathcal{I}(d)$, $\sigma_d(r) := r^\mathcal{I}(d)$ for $r \in \mathsf{N_R}$ and $\sigma_d(C) := C^\mathcal{I} \cap \mathsf{ars}^\mathcal{I}(d)$ for concepts C. The mapping $\cdot^\mathcal{I}$ is extended to succ-restrictions by defining $d \in \mathsf{succ}(\mathtt{con})^\mathcal{I}$ iff σ_d is a solution of `con`.

$\mathcal{ALCOSCC}$ does not need existential restrictions $\exists r.C$, as they can be expressed as $\mathsf{succ}(|r \cap C| \geq 1)$. Moreover, succ-restrictions can compare quantities of successors, e.g. $\mathsf{succ}(|\mathsf{own} \cap \mathsf{Car}| = |\mathsf{child} \cap \mathsf{Human}|)$ describes people who own as many cars as they have children, without specifying the exact amount.

To integrate the concrete domain $\mathfrak{D}$, we complement $\mathsf{N_C}$, $\mathsf{N_R}$ and $\mathsf{N_I}$ with an at most countable set $\mathsf{N_F}$ of *feature names* that connect individuals with values in D [8]. A *feature path* p is of the form f or rf with $r \in \mathsf{N_R}$ and $f \in \mathsf{N_F}$. For instance, salary is a feature name as well as a feature path, while $\mathsf{child\ salary}$ is a feature path including the role name child. *Concrete domain restrictions* (or *CD-restrictions*) are concepts $\exists p_1, \ldots, p_k.P$, where p_i are feature paths and P is a k-ary predicate of $\mathfrak{D}$. An interpretation $\mathcal{I}$ assigns to each $f \in \mathsf{N_F}$ a *partial* function $f^\mathcal{I} \colon \Delta^\mathcal{I} \rightharpoonup D$. A feature path p is mapped to $p^\mathcal{I} \subseteq \Delta^\mathcal{I} \times D$ by defining $p^\mathcal{I}(d) := \{f^\mathcal{I}(d)\}$ if $p = f$[8] and $p^\mathcal{I}(d) := \{f^\mathcal{I}(e) \mid e \in r^\mathcal{I}(d)\}$ if $p = rf$. Then we can define

$$(\exists p_1, \ldots, p_k.P)^\mathcal{I} := \{d \in \Delta^\mathcal{I} \mid \textit{some} \text{ tuple in } p_1^\mathcal{I}(d) \times \cdots \times p_k^\mathcal{I}(d) \text{ is in } P^D\}.$$

For example, one can describe individuals whose salaries are greater than that of some of their children using $\exists \mathsf{salary}, \mathsf{child\ salary}.>$. Furthermore, due to JEPD, we can encode *universal CD-restrictions* $\forall p_1, \ldots, p_k.P$ using the conjunction of all concepts $\neg\exists p_1, \ldots, p_k.P'$ where $P' \neq P$ is a k-ary predicate of $\mathfrak{D}$ [27].

A naive extension of $\mathcal{ALCOSCC}$ with concrete domain reasoning that simply combines succ- and CD-restrictions offers limited expressive power. To improve that, we introduce *feature pointers* α of the form f or $\mathtt{next}\, f$ with $f \in \mathsf{N_F}$ and define *feature roles* $\gamma := P(\alpha_1, \ldots, \alpha_k)$, where each α_i is a feature pointer and P is a k-ary predicate of $\mathfrak{D}$. For example, salary is a pointer to the salary of a given individual d, while $\mathtt{next}\,\mathsf{salary}$ is a pointer to the salary of an individual e that we want to compare to d; the feature role $(\mathsf{salary} < \mathtt{next}\,\mathsf{salary})$ describes a binary relation that contains (d, e) iff the salary of d is smaller than that of e.

We define $\mathcal{ALCOSCC}(\mathfrak{D})$ as the extension of $\mathcal{ALCOSCC}$ with CD-restrictions and succ-restrictions $\mathsf{succ}(\mathfrak{con})$ where $\mathfrak{con}$ can also contain feature roles as set variables. We can now describe individuals that earn less than the majority of their children by

$$C_{\mathsf{ex}} := \mathsf{succ}(|\mathsf{child} \cap (\mathsf{salary} < \mathtt{next}\,\mathsf{salary})| > |\mathsf{child} \cap (\mathsf{salary} < \mathtt{next}\,\mathsf{salary})^c|).$$

Feature roles $\gamma := P(\alpha_1, \ldots, \alpha_k)$ are mapped by interpretations $\mathcal{I}$ to relations $\gamma^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ such that $(d, e) \in \gamma^\mathcal{I}$ iff $(c_1, \ldots, c_k) \in P^D$, where $c_i := f_i^\mathcal{I}(d)$ if $\alpha_i = f_i$ and $c_i := f_i^\mathcal{I}(e)$ if $\alpha_i = \mathtt{next}\, f_i$. The QFBAPA assignment σ_d is extended to map feature roles γ to $\gamma^\mathcal{I} \cap \mathsf{ars}^\mathcal{I}(d)$, and $\mathsf{succ}(\mathfrak{con})^\mathcal{I}$ is defined as before.

An $\mathcal{ALCOSCC}(\mathfrak{D})$ *TBox* $\mathcal{T}$ is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$ between concepts C, D. For example, we can describe an individual $\mathtt{Jane}$ that earns more than $\mathtt{Sam}$, where the role $\mathsf{ref}_{\mathtt{Sam}}$ always points to $\mathtt{Sam}$:

$$\mathcal{T}_{\mathsf{ex}} := \{\, \top \sqsubseteq \mathsf{succ}(\mathsf{ref}_{\mathtt{Sam}} = \{\mathtt{Sam}\}),\ \{\mathtt{Jane}\} \sqsubseteq \exists \mathsf{salary}, \mathsf{ref}_{\mathtt{Sam}}\ \mathsf{salary}.> \,\}.$$

A finitely branching interpretation $\mathcal{I}$ is a *model* of $\mathcal{T}$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$ holds for every CI $C \sqsubseteq D$ in $\mathcal{T}$. A TBox $\mathcal{T}$ is *consistent* if it has a model.

[8] In a slight abuse of notation, we view $f^\mathcal{I}(d)$ both as a value and as a singleton set.

We mentioned above that feature roles make $\mathcal{ALCOSCC}(\mathfrak{D})$ more expressive. Precisely, we can show that some concepts with feature roles cannot be equivalently expressed by only using feature names in CD-restrictions; two concepts are *equivalent* if they are always interpreted by the same sets of individuals.

Theorem 1. *There is no $\mathcal{ALCOSCC}(\mathfrak{Q})$ concept without feature roles that is equivalent to C_{ex}.*

Proof. Assume that there is an $\mathcal{ALCOSCC}(\mathfrak{Q})$ concept D without feature roles that is equivalent to C_{ex}. Consider the interpretation $\mathcal{I}$, where d has salary 0 and five child-successors, two whose salary is 0 and three whose salary is 1. Then, $d \in C_{\mathsf{ex}}^{\mathcal{I}} = D^{\mathcal{I}}$. Construct $\mathcal{J}$ from $\mathcal{I}$ by changing the salary of one of the successors from 1 to 0. Since every individual in $\mathcal{J}$ satisfies the same CD-restrictions, concept names and succ-restrictions without feature roles as in $\mathcal{I}$, we deduce that $d \in D^{\mathcal{J}}$. However, d has more successors with equal salary in $\mathcal{J}$ than successors with greater salary, hence $d \notin C_{\mathsf{ex}}^{\mathcal{J}} = D^{\mathcal{J}}$ must hold, which is a contradiction. □

4 Deciding Consistency

Let $\mathfrak{D}$ be an ExpTime-ω-admissible concrete domain and $\mathcal{T}$ an $\mathcal{ALCOSCC}(\mathfrak{D})$ TBox. In this section, we assume w.l.o.g. that $\mathsf{N_C}$, $\mathsf{N_R}$, $\mathsf{N_I}$ and $\mathsf{N_F}$ are finite and contain exactly the names occurring in $\mathcal{T}$ and that there is at least one individual name; indeed, $\mathcal{T}$ is consistent iff $\mathcal{T} \cup \{\{a\} \sqsubseteq \{a\}\}$ is consistent, where a is a fresh individual name. We define the notion of *individual types*, describing sets of equivalent individual names in an interpretation.

Definition 2. *An* individual type $\mathfrak{a}$ *w.r.t.* $\mathsf{N_I}$ *is a non-empty subset of* $\mathsf{N_I}$*, and a set of individual types* $\mathbb{I}$ *is an* individual type system *for* $\mathsf{N_I}$ *if* $\mathbb{I}$ *partitions* $\mathsf{N_I}$*. Given an interpretation* $\mathcal{I}$*, an individual* $d \in \Delta^{\mathcal{I}}$ has individual type $\mathfrak{a}_{\mathcal{I}}(d) := \{a \in \mathsf{N_I} \mid a^{\mathcal{I}} = d\}$ *if this set is non-empty, and* d *is* anonymous *otherwise.*

We now fix an individual type system $\mathbb{I}$. Let $\mathcal{M}$ be the set of all subconcepts appearing in $\mathcal{T}$, as well as their negations. We define the notion of *type* as usual.

Definition 3. *A* type *w.r.t.* $\mathcal{T}$ *is a set* $t \subseteq \mathcal{M}$ *such that:*

- *if* $C \sqsubseteq D \in \mathcal{T}$ *and* $C \in t$*, then* $D \in t$*;*
- *if* $\neg C \in \mathcal{M}$*, then* $C \in t$ *iff* $\neg C \notin t$*;*
- *if* $C \sqcap C' \in \mathcal{M}$*, then* $C \sqcap C' \in t$ *iff* $C \in t$ *and* $C' \in t$*.*

The type *of* $d \in \Delta^{\mathcal{I}}$ *w.r.t.* $\mathcal{T}$ *is the set* $t_{\mathcal{I}}(d) := \{C \in \mathcal{M} \mid d \in C^{\mathcal{I}}\}$*.*

If $\mathcal{I}$ is a model of $\mathcal{T}$, then $t_{\mathcal{I}}(d)$ is indeed a type w.r.t. $\mathcal{T}$. A type t is *named with* an individual type $\mathfrak{a}_t$ if for all $a \in \mathsf{N_I}$, $a \in \mathfrak{a}_t$ iff $\{a\} \in t$, and is *anonymous* if it is not named with any individual type.

Following the approach used in [3], we construct a QFBAPA formula ϕ_t that is induced by the succ-restrictions $\mathsf{succ}(\mathfrak{con})$ in a type t and enriched with constraints derived from the individual type system $\mathbb{I}$ and the set of role names $\mathsf{N_R}$. Formally, ϕ_t is defined as the conjunction of

- $\phi_{\mathfrak{con}}$ if $\mathsf{succ}(\mathfrak{con}) \in t$ and $\neg\phi_{\mathfrak{con}}$ otherwise, where $\phi_{\mathfrak{con}}$ is derived from $\mathfrak{con}$ by replacing role names r, feature roles γ and concepts C with set variables X_r, X_γ and X_C, respectively;
- $|\bigcap_{a\in\mathfrak{a}} X_{\{a\}}| \leq 1$ for every $\mathfrak{a} \in \mathbb{I}$; and
- $\mathcal{U} = \bigcup_{r\in\mathsf{N_R}} X_r$.

All formulae ϕ_t contain exactly the same set variables and thus have the same Venn regions (cf. Section 2), called the *Venn regions of* $\mathcal{T}$. A Venn region v of $\mathcal{T}$ *has individual type* $\mathfrak{a}_v = \{a \in \mathsf{N_I} \mid X_{\{a\}} \in v\}$ if this set is non-empty, and v is *anonymous* otherwise. The following example shows that ϕ_t does not yet account for the numerical constraints induced by the CD-restrictions in t.

Example 1. Let $\mathcal{T} = \{\, \top \sqsubseteq (\exists \mathsf{salary}, \mathsf{child\ salary}.<) \sqcap (\mathsf{succ}(|\mathsf{child}| \leq 0)) \,\}$. For every model $\mathcal{I}$ of $\mathcal{T}$ and $d \in \Delta^{\mathcal{I}}$, the type $t := t_{\mathcal{I}}(d)$ contains both conjuncts appearing in this CI. The QFBAPA formula $\phi_t := |X_{\mathsf{child}}| \leq 0 \wedge \mathcal{U} = X_{\mathsf{child}}$ is satisfied by any solution assigning the empty set to $\mathcal{U}$. However, t cannot be realized: the first conjunct implies that d has a child-successor $e \neq d$ such that $\mathsf{salary}^{\mathcal{I}}(d) < \mathsf{salary}^{\mathcal{I}}(e)$, while the last conjunct forces d to have no child-successor.

To realize the CD-restrictions in t, we may need up to $M_{\mathcal{T}} := R_{\mathcal{T}} \cdot P_{\mathcal{T}}$ distinct role successors, where $R_{\mathcal{T}}$ is the number of CD-restrictions in $\mathcal{M}$ and $P_{\mathcal{T}}$ is the maximal arity of predicates of $\mathfrak{D}$ occurring in $\mathcal{M}$. We add this information to the QFBAPA formula ϕ_t with additional constraints over a set of pre-selected Venn regions, representing sets of role successors whose existence is implied by the CD-restrictions in t. Let S be a set of at most $M_{\mathcal{T}}$ Venn regions v, each associated to a natural number $0 \leq n_v \leq M_{\mathcal{T}}$. By Lemma 1, the QFBAPA formula $\phi_{t,S}$, which extends ϕ_t with a conjunct $|v| \geq n_v$ for each $v \in S$, is satisfiable iff there is a natural number $N_{\mathcal{T}}$ of polynomial size w.r.t. the size of ϕ_t and $M_{\mathcal{T}}$ s.t. $\phi_{t,S}$ has a solution in which at most $N_{\mathcal{T}}$ Venn regions are non-empty. Moreover, since all formulae ϕ_t are nearly of the same size (except for the difference between $\phi_{\mathfrak{con}}$ and $\neg\phi_{\mathfrak{con}}$) and $|S|$ and the numbers n_v are bounded by $M_{\mathcal{T}}$, we can assume that the bound $N_{\mathcal{T}}$ is independent of the choice of S and t, is polynomial w.r.t. the size of $\mathcal{T}$ and can be computed in polynomial time.

To formalize these additional restrictions, we consider *bags*, i.e. functions V assigning to every Venn region v of $\mathcal{T}$ a *multiplicity* $V(v) \in \mathbb{N}$, whose *support* $\mathsf{supp}(V)$ is the set of Venn regions of $\mathcal{T}$ with multiplicity $V(v) \geq 1$. The associated QFBAPA formula ϕ_V is the conjunction of the constraint $\mathcal{U} = \bigcup \mathsf{supp}(V)$ and all constraints $|v| \geq c$ where $v \in \mathsf{supp}(V)$ and $c = V(v)$.

Definition 4. *A* Venn bag *for a type t w.r.t. $\mathcal{T}$ is a bag V of Venn regions of $\mathcal{T}$ s.t. $|\mathsf{supp}(V)| \leq N_{\mathcal{T}}$, $V(v) \leq M_{\mathcal{T}}+1$ holds for all $v \in \mathsf{supp}(V)$ and the QFBAPA formula $\phi_{t,V} := \phi_t \wedge \phi_V$ is satisfiable.*

By Lemma 1, $\phi_{t,S}$ is satisfiable iff there is a Venn bag V for t such that $\phi_{t,V}$ includes all constraints from $\phi_{t,S}$.

Finally, we take care of actually satisfying the CD-restrictions occurring in a type by using complete constraint systems to describe all relevant feature values. Feature values that are not represented in these systems correspond to undefined values. To ensure that all types agree on the feature values of individual names, we fix an *individual constraint system* $\mathfrak{C}_{\mathbb{I}}$ w.r.t. $\mathbb{I}$, i.e. a complete constraint system over variables of the form $f^{\mathfrak{a}}$, where $f \in \mathsf{N_F}$ and $\mathfrak{a} \in \mathbb{I}$, that refer to the feature values of named individuals. Then, we define constraint systems $\mathfrak{C}_{t,V}$ representing the relations between the feature values associated with an individual of type t and those of its role successors as specified by a Venn bag V for t. The system $\mathfrak{C}_{t,V}$ extends $\mathfrak{C}_{\mathbb{I}}$ by adding variables of the form

- $f^\star$, representing the value of the feature $f \in \mathsf{N_F}$ at the current individual;
- $f^{(v,j)}$ with $v \in \mathsf{supp}(V)$ and $1 \leq j \leq V(v)$ for the f-values at the successors, in order to express the relevant CD-restrictions.

Again, not all these variables actually need to occur in the constraint system, only the ones whose associated feature values should be defined. To handle named types and named Venn regions, we define the indexing functions

$$\iota(t) := \begin{cases} \star & \text{if } t \text{ is anonymous} \\ \mathfrak{a}_t & \text{otherwise} \end{cases} \qquad \text{and } \iota((v,j)) := \begin{cases} (v,j) & \text{if } v \text{ is anonymous} \\ \mathfrak{a}_v & \text{otherwise} \end{cases}$$

for all $v \in \mathsf{supp}(V)$ and $1 \leq j \leq V(v)$. Additionally, we do not allow more variables of the form $f^{\mathfrak{a}}$ than those already contained in $\mathfrak{C}_{\mathbb{I}}$.

Definition 5. *Let t be a type w.r.t. $\mathcal{T}$ and V a Venn bag for t. A* local system *for t, V is a complete constraint system $\mathfrak{C}_{t,V}$ that includes $\mathfrak{C}_{\mathbb{I}}$ and no additional variables of the form $f^{\mathfrak{a}}$, $\mathfrak{a} \in \mathbb{I}$, such that:*

1. *if $C := \exists p_1, \ldots, p_k.P \in \mathcal{M}$, then $C \in t$ iff $P(f_1^{x_1}, \ldots, f_k^{x_k}) \in \mathfrak{C}_{t,V}$ such that*

$$x_i = \begin{cases} \iota(t) & \textit{if } p_i = f_i, \textit{ or} \\ \iota((v,j)) & \textit{if } p_i = rf_i, \textit{ for some } 1 \leq j \leq V(v) \textit{ and } X_r \in v; \end{cases}$$

2. *for all set variables $X_{P(\alpha_1,\ldots,\alpha_k)}$, all $v \in \mathsf{supp}(V)$, and $1 \leq j \leq V(v)$ it holds that $X_{P(\alpha_1,\ldots,\alpha_k)} \in v$ iff $P(f_1^{x_1}, \ldots, f_k^{x_k}) \in \mathfrak{C}_{t,V}$, where*

$$x_i = \begin{cases} \iota(t) & \textit{if } \alpha_i = f_i, \textit{ and} \\ \iota((v,j)) & \textit{if } \alpha_i = \texttt{next}\, f_i. \end{cases}$$

In the following definition, we denote with S_v the subset of $\mathcal{M}$ that contains $C \in \mathcal{M}$ if $X_C \in v$ and $\neg C \in \mathcal{M}$ if $X_C \notin v$ (cf. Section 2).

Definition 6. *An* augmented type *for $\mathcal{T}$ is a tuple $\mathfrak{t} := (t, V, \mathfrak{C}_{\mathfrak{t}})$, where t is a type w.r.t. $\mathcal{T}$, V is a Venn bag for t, and $\mathfrak{C}_{\mathfrak{t}}$ is a satisfiable local system for t, V. The* root *of $\mathfrak{t}$ is $\mathsf{root}(\mathfrak{t}) := t$.*

An augmented type $\mathfrak{t}' = (t', V', \mathfrak{C}_{\mathfrak{t}'})$ patches *$\mathfrak{t}$ at (v, i), where $v \in \mathsf{supp}(V)$ and $1 \leq i \leq V(v)$, if $S_v \subseteq t'$ and the* merged system *$\mathfrak{C}_{\mathfrak{t}} \lhd_{(v,i)} \mathfrak{C}_{\mathfrak{t}'}$ has a solution,*

Algorithm 1 Type elimination algorithm for $\mathcal{ALCOSCC}(\mathfrak{D})$

Input: An $\mathcal{ALCOSCC}(\mathfrak{D})$ TBox $\mathcal{T}$
Output: CONSISTENT if $\mathcal{T}$ is consistent, and INCONSISTENT otherwise
1: **guess** an individual type system $\mathbb{I}$ and an individual constraint system $\mathfrak{C}_{\mathbb{I}}$
2: **guess** augmented types $\mathfrak{t}_{\mathfrak{a}} = (t_{\mathfrak{a}}, V_{\mathfrak{a}}, \mathfrak{C}_{\mathfrak{a}})$ for $\mathfrak{a} \in \mathbb{I}$ s.t. $t_{\mathfrak{a}}$ is named with $\mathfrak{a}$
3: $\mathbb{T} \leftarrow \{\mathfrak{t} = (t, V, \mathfrak{C})$ augmented type $\mid t$ is anonymous$\} \cup \{\mathfrak{t}_{\mathfrak{a}} \mid \mathfrak{a} \in \mathbb{I}\}$
4: **while** there is $\mathfrak{t} \in \mathbb{T}$ that is not patched by $\mathbb{T}$ **do** $\mathbb{T} \leftarrow \mathbb{T} \setminus \{t\}$
5: **if** $\mathfrak{t}_{\mathfrak{a}} \in \mathbb{T}$ for all $\mathfrak{a} \in \mathbb{I}$ **then return** CONSISTENT
6: **else return** INCONSISTENT

where $\mathfrak{C}_{\mathfrak{t}} \lhd_{(v,i)} \mathfrak{C}_{\mathfrak{t}'}$ is obtained as the union of $\mathfrak{C}_{\mathfrak{t}}$ and the result of replacing all variables in $\mathfrak{C}_{\mathfrak{t}'}$ as follows:

$$\begin{array}{lll} f^\star & \mapsto f^{(v,i)} & \text{if } t' \text{ is anonymous;} \\ f^{(w,j)} & \mapsto f^{(w,j)'} & \text{for all anonymous } w \in \mathsf{supp}(V') \text{ and } 1 \leq j \leq V'(w); \\ f^{\mathfrak{a}} & \mapsto f^{\mathfrak{a}} & \text{for all } \mathfrak{a} \in \mathbb{I}. \end{array}$$

A set of augmented types $\mathbb{T}$ patches $\mathfrak{t}$ *if, for all $v \in \mathsf{supp}(V)$ and $1 \leq i \leq V(v)$, there is a $\mathfrak{t}' \in \mathbb{T}$ that patches $\mathfrak{t}$ at (v, i).*

The merging operation identifies all features associated to (v, i) in $\mathfrak{C}_{\mathfrak{t}}$ with those associated to t' in $\mathfrak{C}_{\mathfrak{t}'}$, while keeping the remaining variables associated to anonymous individuals separate. If t' is not anonymous (and thus $\mathfrak{C}_{\mathfrak{t}'}$ contains no variable of the form $f^\star$) then the condition $S_v \subseteq t'$ ensures that $\mathfrak{a}_v = \mathfrak{a}_{t'}$, and thus the variable $f^{\iota((v,i))} = f^{\mathfrak{a}_v} = f^{\mathfrak{a}_{t'}}$ in $\mathfrak{C}_{\mathfrak{t}}$ is already identical to $f^{\iota(t')} = f^{\mathfrak{a}_{t'}}$ in $\mathfrak{C}_{\mathfrak{t}'}$.

The augmented types are now used by Algorithm 1 to decide consistency of an $\mathcal{ALCOSCC}(\mathfrak{D})$ TBox via a type elimination approach. We show that Algorithm 1 is indeed sound and complete.

Lemma 2. *If there is a run of Algorithm 1 that returns* CONSISTENT, *then $\mathcal{T}$ is consistent.*

Proof. We construct a model $\mathcal{I}$ of $\mathcal{T}$ using the individual type system $\mathbb{I}$ and the set $\mathbb{T}$ of augmented types constructed by Algorithm 1. The domain $\Delta^{\mathcal{I}}$ consists of tuples $(\mathfrak{a}, w)$, where $\mathfrak{a} \in \mathbb{I}$ and w is a word over the alphabet Σ of all tuples $(\mathfrak{t}, v, i)$ with $\mathfrak{t} \in \mathbb{T}$, v a Venn region of $\mathcal{T}$ and $i \geq 1$ a natural number. We associate to each tuple $(\mathfrak{a}, w)$ the augmented type $\mathsf{end}(\mathfrak{a}, w) \in \mathbb{T}$ defined as $\mathsf{end}(\mathfrak{a}, \varepsilon) := \mathfrak{t}_{\mathfrak{a}}$ and $\mathsf{end}(\mathfrak{a}, w' \cdot (\mathfrak{t}, v, i)) := \mathfrak{t}$ for $w' \in \Sigma^*$.

We define $\Delta^{\mathcal{I}}$ as the union of sets Δ^m with $m \in \mathbb{N}$, where Δ^0 contains $(\mathfrak{a}, \varepsilon)$ for every $\mathfrak{a} \in \mathbb{I}$ and Δ^{m+1} is defined inductively in the following. Given $(\mathfrak{a}, w) \in \Delta^m$ with $\mathsf{end}(\mathfrak{a}, w) = \mathfrak{t} = (t, V, \mathfrak{C}_{t,V}) \in \mathbb{T}$ we observe that

- the QFBAPA formula ϕ_t has a solution $\sigma_{\mathfrak{a},w}$ such that $\sigma_{\mathfrak{a},w}(|v|) \geq V(v)$ if $v \in \mathsf{supp}(V)$ and $\sigma_{\mathfrak{a},w}(|v|) = 0$ otherwise for all Venn regions v of $\mathcal{T}$,
- for $v \in \mathsf{supp}(V)$ and $i = 1, \ldots, V(v)$ there exists an augmented type $\mathfrak{t}_{(v,i)} \in \mathbb{T}$ patching $\mathfrak{t}$ at (v, i), as otherwise $\mathfrak{t}$ would have been eliminated from $\mathbb{T}$.

Using these augmented types, we define for $r \in \mathsf{N_R}$ the set $\Delta_r^{m+1}[\mathfrak{a}, w]$ containing $(\mathfrak{a}, w \cdot (\mathfrak{t}_{(v,i)}, v, j))$ iff $X_r \in v$, $\mathsf{root}(\mathfrak{t}_{(v,i)})$ is anonymous, $j = 1, \ldots, \sigma_{\mathfrak{a},w}(|v|)$ and $i = \max(j, V(v))$. We now define Δ^{m+1} as the extension of Δ^m by all sets $\Delta_r^{m+1}[\mathfrak{a}, w]$ for which $r \in \mathsf{N_R}$ and $(\mathfrak{a}, w) \in \Delta^m$.

The extensions of $a \in \mathsf{N_I}$, $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$ are defined as:

$$a^{\mathcal{I}} := (\mathfrak{a}, \varepsilon), \text{ where } \mathfrak{a} \text{ is the unique individual type in } \mathbb{I} \text{ containing } a;$$

$$A^{\mathcal{I}} := \{(\mathfrak{a}, w) \in \Delta^{\mathcal{I}} \mid \mathsf{end}(\mathfrak{a}, w) = \mathfrak{t} \text{ and } A \in \mathsf{root}(\mathfrak{t})\};$$

$$r^{\mathcal{I}} := \{((\mathfrak{a}, w), (\mathfrak{b}, \varepsilon)) \mid \mathsf{end}(\mathfrak{a}, w) = \mathfrak{t} \text{ and } \mathfrak{t}_{\mathfrak{b}} \text{ patches } \mathfrak{t} \text{ at } (v, i) \text{ with } X_r \in v\} \cup \{((\mathfrak{a}, w), (\mathfrak{a}, w')) \mid (\mathfrak{a}, w) \in \Delta^m \text{ and } (\mathfrak{a}, w') \in \Delta_r^{m+1}[\mathfrak{a}, w] \text{ with } m \in \mathbb{N}\}.$$

For $f \in \mathsf{N_F}$, $f^{\mathcal{I}}$ is defined as follows. If $(\mathfrak{a}, w) \in \Delta^{\mathcal{I}}$ and $\mathsf{end}(\mathfrak{a}, w) = (t, V, \mathfrak{C}_{t,V})$, we extend $\mathfrak{C}_{t,V}$ with all variables $f^{(v,i)}$ where $i > V(v)$ and $f \in \mathsf{N_F}$ such that $f^{(v,V(v))}$ occurs in $\mathfrak{C}_{t,V}$ and $(\mathfrak{a}, w \cdot (\mathfrak{t}', v, i))$ occurs in $\Delta^{\mathcal{I}}$. Then, we add all constraints $P((f_1^{x_1})', \ldots, (f_k^{x_k})')$ obtained by replacing every occurrence of $f^{(v,V(v))}$ in a constraint $P(f_1^{x_1}, ..., f_k^{x_k}) \in \mathfrak{C}_{t,V}$ with some variable $f^{(v,i)}$ among those occurring in the extended system with $i \geq V(v)$. In this way, the feature values of all role successors of $(\mathfrak{a}, w)$ are handled correctly w.r.t. one another and w.r.t. those of $(\mathfrak{a}, w)$. Next, we replace all variables $f^{\mathfrak{a}}$ with $f^{\mathfrak{a},\varepsilon}$, all variables $f^{\star}$ with $f^{\mathfrak{a},w}$ and all variables $f^{(v,i)}$ with $f^{\mathfrak{a},u}$ where u is the unique word of the form $w \cdot (\mathfrak{t}', v, i)$ for which $(\mathfrak{a}, u) \in \Delta^{\mathcal{I}}$.

Let $\mathfrak{C}_{\mathfrak{a},w}$ be the resulting complete constraint system and $\mathfrak{C}^m$ with $m \in \mathbb{N}$ be the union of all $\mathfrak{C}_{\mathfrak{a},w}$ with $(\mathfrak{a}, w) \in \Delta^m$. We show in [5] (Lemma 6) that for every $m \in \mathbb{N}$, the constraint system $\mathfrak{C}^m$ has a solution. Let $\mathfrak{C}^{\mathcal{I}}$ be the union of all systems $\mathfrak{C}^m$ with $m \in \mathbb{N}$. Every finite subsystem of $\mathfrak{C}^{\mathcal{I}}$ is a subsystem of $\mathfrak{C}^m$ for some $m \in \mathbb{N}$ and is thus satisfiable. Thus, by homomorphism ω-compactness of $\mathfrak{D}$, we can infer that $\mathfrak{C}^{\mathcal{I}}$ has a solution $h^{\mathcal{I}}$. For every $f \in \mathsf{N_F}$ and $(\mathfrak{a}, w) \in \Delta^{\mathcal{I}}$, we now define $f^{\mathcal{I}}((\mathfrak{a}, w)) := h^{\mathcal{I}}(f^{\mathfrak{a},w})$ if $f^{\mathfrak{a},w}$ occurs in $\mathfrak{C}^{\mathcal{I}}$ and leave it undefined otherwise.

We show in [5] (Lemma 7) that for all $d = (\mathfrak{a}, w) \in \Delta^{\mathcal{I}}$ and $C \in \mathcal{M}$, we have $C \in \mathsf{root}(\mathsf{end}(d))$ iff $d \in C^{\mathcal{I}}$. It is a direct consequence that $\mathcal{I}$ satisfies all CIs in $\mathcal{T}$ and is thus a model of $\mathcal{T}$. Hence, $\mathcal{T}$ is consistent. □

Lemma 3. *If $\mathcal{T}$ is consistent, then there is a run of Algorithm 1 that returns CONSISTENT.*

Proof. Let $\mathcal{I}$ be a model of $\mathcal{T}$ and $\mathbb{I}$ be the individual type system that contains an individual type $\mathfrak{a}$ iff $\mathfrak{a} = \mathfrak{a}_{\mathcal{I}}(d)$ for some $d \in \Delta^{\mathcal{I}}$ (cf. Definition 2). Then $\mathbb{I}$ is well-defined, as every $a \in \mathsf{N_I}$ is uniquely assigned to $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. For $\mathfrak{a} \in \mathbb{I}$, we denote by $\mathfrak{a}^{\mathcal{I}}$ the unique $d \in \Delta^{\mathcal{I}}$ with $\mathfrak{a} = \mathfrak{a}_{\mathcal{I}}(d)$. Further, let $T_{\mathcal{I}} := \{t_{\mathcal{I}}(d) \mid d \in \Delta^{\mathcal{I}}\}$ be the set of all types that are realized in $\mathcal{I}$.

For each individual type $\mathfrak{a} \in \mathbb{I}$, we define $t_{\mathfrak{a}} := t_{\mathcal{I}}(\mathfrak{a}^{\mathcal{I}})$. Using $\mathbb{I}$ and $T_{\mathcal{I}}$, we build a constraint system $\mathfrak{C}_{\mathbb{I}}$ and a set $\mathbb{T}_{\mathcal{I}} := \{\mathfrak{t}_{\mathcal{I}}(d) \mid d \in \Delta^{\mathcal{I}}\}$ of augmented types, containing unique types $\mathfrak{t}_{\mathfrak{a}}$ whose roots are named with $\mathfrak{a} \in \mathbb{I}$. We define the individual constraint system $\mathfrak{C}_{\mathbb{I}}$ over all variables $f^{\mathfrak{a}}$ with $f \in \mathsf{N_F}$

and $\mathfrak{a} \in \mathbb{I}$ for which $f^{\mathcal{I}}(\mathfrak{a}^{\mathcal{I}})$ is defined, such that $P(f_1^{\mathfrak{a}_1}, \ldots, f_k^{\mathfrak{a}_k}) \in \mathfrak{C}_{\mathbb{I}}$ iff $\big(f_1^{\mathcal{I}}(\mathfrak{a}_1^{\mathcal{I}}), \ldots, f_k^{\mathcal{I}}(\mathfrak{a}_k^{\mathcal{I}})\big) \in P^D$. Clearly, $\mathfrak{C}_{\mathbb{I}}$ is complete.

Next, we associate to each $d \in \Delta^{\mathcal{I}}$ an augmented type $\mathfrak{t}_{\mathcal{I}}(d) := (t_{\mathcal{I}}(d), V_d, \mathfrak{C}_d)$. If e is a role successor of d, let v_e be the Venn region of $\mathcal{T}$ whose variables X_r, X_C, X_γ for role names r, concepts C and feature roles γ satisfy the following:

- $X_r \in v_e$ iff e is an r-successor of d;
- $X_C \in v_e$ iff $C \in t_{\mathcal{I}}(e)$;
- $X_\gamma \in v_e$ iff $e \in \gamma^{\mathcal{I}}(d)$.

For every non-negated CD-restriction $\exists p_1, \ldots, p_k.P \in t_{\mathcal{I}}(d)$, we can find values $c_i \in p_i^{\mathcal{I}}(d)$ for $i = 1, \ldots, k$ such that $(c_1, \ldots, c_k) \in P^D$. If $p_i = r_i f_i$, this implies that there is $e_i \in r_i^{\mathcal{I}}(d)$ such that $f_i^{\mathcal{I}}(e_i) = c_i$. We collect all these successors of d, which are at most $M_{\mathcal{T}}$ many distinct elements, in the set S_{cd}. For $e \in \mathrm{S}_{\mathsf{cd}}$, let n_e be the number of elements $e' \in \mathrm{S}_{\mathsf{cd}}$ such that $v_e = v_{e'}$. We show in [5] (Lemma 8) that there is a Venn bag V_d for $t_{\mathcal{I}}(d)$ w.r.t. $\mathcal{T}$ such that $V_d(v_e) = n_e$ for all $e \in \mathrm{S}_{\mathsf{cd}}$, and for all other $v \in \mathsf{supp}(V_d)$ we have $V_d(v) = 1$ and there is a role successor $e \in \Delta^{\mathcal{I}} \setminus \mathrm{S}_{\mathsf{cd}}$ of d such that $v = v_e$.

It remains to define the local system $\mathfrak{C}_d$. Consider the set X_d that contains $\iota(t_{\mathcal{I}}(d))$ (either $\star$ or $\mathfrak{a}_{t_{\mathcal{I}}(d)}$), all $\mathfrak{a} \in \mathbb{I}$, and all pairs (v, j) with anonymous Venn bags $v \in \mathsf{supp}(V_d)$ and $1 \leq j \leq V_d(v)$ (cf. Definition 5). Let λ_d be a bijection mapping every such (v, j) to an anonymous successor e of d satisfying $v_e = v$, such that $\lambda_d((v_e, j)) \in \mathrm{S}_{\mathsf{cd}}$ for all $e \in \mathrm{S}_{\mathsf{cd}}$. Such a bijection exists due to Lemma 8 in [5]. We extend this bijection to $\mathbb{I}$ by setting $\lambda_d(\mathfrak{a}) := \mathfrak{a}^{\mathcal{I}}$. Furthermore, we extend λ_d to ξ_d with $\xi_d(\iota(t_{\mathcal{I}}(d))) := d$. Then, ξ_d is injective except for $\star$: if d is its own role successor, it can happen that $\xi_d(\star) = d = \xi_d((v, j))$. We define the complete constraint system $\mathfrak{C}_d$ over variables f^x with $f \in \mathsf{N_F}$, $x \in X_d$, such that $P(f_1^{x_1}, \ldots, f_k^{x_k}) \in \mathfrak{C}_d$ iff $\big(f_1^{\mathcal{I}}(\xi_d(x_1)), \ldots, f_k^{\mathcal{I}}(\xi_d(x_k))\big) \in P^D$ holds for all $x_1, \ldots, x_k \in X_d$. If $f^{\mathcal{I}}(\xi_d(x))$ is undefined, then f^x does not occur in $\mathfrak{C}_d$.

We show in [5] (Lemma 9) that $\mathfrak{C}_d$ is a satisfiable local system for $t_{\mathcal{I}}(d)$ and V_d. Thus, $\mathfrak{t}_{\mathcal{I}}(d) = \big(t_{\mathcal{I}}(d), V_d, \mathfrak{C}_d\big)$ is an augmented type. Furthermore, we can show (cf. Lemma 10 in [5]) that every augmented type in $\mathbb{T}_{\mathcal{I}}$ is patched in $\mathbb{T}_{\mathcal{I}}$. If Algorithm 1 guesses $\mathbb{I}$, $\mathfrak{C}_{\mathbb{I}}$, and $\mathfrak{t}_{\mathfrak{a}}$, where $\mathfrak{a} \in \mathbb{I}$, then the initial set $\mathbb{T}$ must contain $\mathbb{T}_{\mathcal{I}}$, and no augmented type from $\mathbb{T}_{\mathcal{I}}$ can ever be removed from $\mathbb{T}$. This shows that the augmented types $\{\mathfrak{t}_{\mathfrak{a}} \mid \mathfrak{a} \in \mathsf{N_I}\} \subseteq \mathbb{T}_{\mathcal{I}}$ remain in $\mathbb{T}$ throughout the execution of the algorithm. Since the algorithm terminates, it thus returns CONSISTENT. $\square$

Because Algorithm 1 runs in exponential time, we obtain a matching upper bound to the ExpTime-hardness inherited from $\mathcal{ALC}$. Indeed, as there are at most exponentially many individual type systems and polynomially many individual types in such a type system, all guesses can be implemented by enumerating all choices in exponential time. The main elimination procedure also runs in exponential time as the number of augmented types is exponentially bounded, and all required tests can be performed in exponential time, provided that $\mathfrak{D}$ is ExpTime-ω-admissible. We thus obtain the following result.

Theorem 2. *Let* $\mathfrak{D}$ *be an* ExpTime-ω-admissible concrete domain. Then, consistency checking in $\mathcal{ALCOSCC}(\mathfrak{D})$ is ExpTime-complete.

5 Reasoning with ABoxes

In DLs, a TBox is often complemented by an *ABox* containing *concept assertions* $C(a)$ and *role assertions* $r(a, b)$, where $a, b \in \mathsf{N_I}$, $r \in \mathsf{N_R}$, and C is a concept, with the obvious semantics. In our DL, those assertions can be expressed in the TBox using nominals [9]. In the presence of a concrete domain, however, we may want to use additional kinds of assertions: *predicate assertions* $P(f_1(a_1), \dots, f_k(a_k))$ with $f_i \in \mathsf{N_F}$, $a_i \in \mathsf{N_I}$, $i = 1, \dots, k$, and a k-ary predicate P of $\mathfrak{D}$, and *feature assertions* $f(a, c)$ with $f \in \mathsf{N_F}$, $a \in \mathsf{N_I}$, and a constant $c \in D$. The former requires every model $\mathcal{I}$ to satisfy $(f_1^{\mathcal{I}}(a_1^{\mathcal{I}}), \dots, f_k^{\mathcal{I}}(a_k^{\mathcal{I}})) \in P^D$, and the latter states that $f^{\mathcal{I}}(a^{\mathcal{I}}) = c$.

Using predicate assertions, we can rewrite the TBox $\mathcal{T}_{\mathsf{ex}}$ in Section 3 into a single, intuitive assertion $\mathsf{salary}(\mathtt{Sam}) < \mathsf{salary}(\mathtt{Jane})$. This also demonstrates how predicate assertions can be simulated by CIs: instead of $P(f_1(a_1), \dots, f_k(a_k))$, we can use $\top \sqsubseteq \mathsf{succ}(\mathsf{ref}_{a_i} = \{a_i\})$ for $i = 1, \dots, k$ and $\top \sqsubseteq \exists \mathsf{ref}_{a_1} f_1, \dots, \mathsf{ref}_{a_k} f_k.P$.

On the other hand, with feature assertions, we can give specific values and state, for instance, that $\mathtt{Sam}$'s salary is 100,001 € with $\mathsf{salary}(\mathtt{Sam}, 100{,}001)$. Feature assertions seemingly increase the expressivity, since we can use them to refer to constant values. However, we first have to specify how these constants are actually encoded. For the following results, we consider *concrete domains* $\mathfrak{D}$ *with constants*, which extend concrete domains with an encoding for arbitrary constants $c \in D$ and constraint systems that can use such constants in addition to variables. For ExpTime-ω-admissible concrete domains with constants, we also require the extended CSP($\mathfrak{D}$) to be decidable in exponential time. In particular, the main known examples of ExpTime-ω-admissible concrete domains ($\mathfrak{Q}$, Allen's relations, and RCC8) satisfy this requirement under the reasonable assumptions that all numbers are given as integer fractions in binary encoding and the constants in RCC8 refer to polygonal regions in the rational plane [21,24]. We can now use this encoding to represent the constants in feature assertions $f(a, c)$.

Unfortunately, we cannot directly use constants in constraints to extend Algorithm 1 to support feature assertions, since this would result in infinitely many possible local constraint systems. Another idea to deal with feature assertions is that, if $\mathfrak{D}$ has *singleton predicates* $=_c$ with $(=_c)^D = \{c\}$, then one can express $f(a, c)$ by $\{a\} \sqsubseteq \exists f.{=_c}$. Since an ω-admissible concrete domain $\mathfrak{D}$ has a finite signature, however, this only works for a fixed, finite set of values $c \in D$. Due to the JD and JEPD conditions, it turns out that feature assertions are actually equivalent to *additional singleton predicates* $=_c$ that are not part of $\mathfrak{D}$, but can be used in concepts with the same semantics as defined above.[9]

[9] The results in this section also hold for $\mathcal{ALC}(\mathfrak{D})$, since they can be shown without succ-restrictions, nominals, or restricting to finitely-branching interpretations [13].

Lemma 4. *For $\mathcal{ALCOSCC}(\mathfrak{D})$ with an ω-admissible concrete domain $\mathfrak{D}$ with constants, the following problems are reducible to each other: (a) consistency with additional singleton predicates, and (b) consistency with feature assertions. The reductions take exponential time, but produce ontologies of polynomial size.*

Additionally, feature assertions can be expressed by *predicate assertions* if $\mathfrak{D}$ is *homogeneous*, i.e. such that every isomorphism between finite substructures of $\mathfrak{D}$ can be extended to an isomorphism from $\mathfrak{D}$ to itself [11]. All known ω-admissible concrete domains are homogeneous [11].

Lemma 5. *For $\mathcal{ALCOSCC}(\mathfrak{D})$ with an ω-admissible and homogeneous concrete domain $\mathfrak{D}$ with constants, consistency with feature assertions can be reduced to consistency without feature assertions in exponential time. The resulting ontology is of polynomial size.*

Together, Lemmas 4 to 5 show that, under these conditions, we can freely use constant values (either in feature assertions or additional singleton predicates) in $\mathcal{ALCOSCC}(\mathfrak{D})$, without increasing the complexity of reasoning. The following result then follows together with Theorem 2.

Theorem 3. *If $\mathfrak{D}$ is an* ExpTime-ω-admissible and homogeneous concrete domain with constants, then consistency in $\mathcal{ALCOSCC}(\mathfrak{D})$ with feature assertions and additional singleton predicates is ExpTime-complete.

6 Undecidable Extensions

To conclude our investigations, we show that several extensions of $\mathcal{ALCOSCC}(\mathfrak{D})$, inspired by existing DLs or obtained by seemingly harmless tweaks to the syntax and semantics, are undecidable. Hereafter, we assume that the domain set of $\mathfrak{D}$ is infinite and that $\mathfrak{D}$ is JD (cf. Section 2). In this case, we allow w.l.o.g. the usage of set terms $(f = \texttt{next}\, g)$ expressing equality of the values assigned to f and $\texttt{next}\, g$ (we detail the construction of this term in [5]).

Comparing set cardinalities and feature values. If $\mathfrak{D}$ is a numerical concrete domain where D is either $\mathbb{N}$, $\mathbb{Z}$ or $\mathbb{Q}$, it is natural to consider comparisons between feature values of an individual d and the cardinalities of sets of role successors of d. For example, we could describe individuals whose age is twice the number of their children using the concept $\mathsf{succ}(\mathsf{age} = 2 \cdot |\mathsf{child}|)$. This could be achieved by allowing succ-restrictions to contain *mixed numerical constraints* $f = \ell$, where ℓ is a PA expression (cf. Section 2) that is allowed in $\mathcal{ALCOSCC}(\mathfrak{D})$ and $f \in \mathsf{N_F}$; then, we extend $\cdot^{\mathcal{I}}$ by defining $d \in \mathsf{succ}(f = \ell)^{\mathcal{I}}$ iff $f^{\mathcal{I}}(d) = \sigma_d(\ell)$. Unfortunately, for the CDs considered here, this leads to undecidability, which can be shown by a reduction to $\mathcal{ALC}(\mathfrak{D})$ with the concrete domain $\mathfrak{D} = (\mathbb{N}, +_1)$ where $+_1$ is the successor relation, which is known to be undecidable [11].

Theorem 4. *If $\mathfrak{D}$ is a numerical concrete domain that is JD, then consistency of $\mathcal{ALCOSCC}(\mathfrak{D})$ TBoxes with mixed numerical constraints is undecidable.*

Proof. We force $r \in \mathsf{N_R}$ to be functional with the CI $\top \sqsubseteq \mathsf{succ}(|r| \leq 1)$. We encode the CD-restriction $C := \exists p_0, p_1.+_1$ using $C_0 \sqcap C_1$, where

$$C_i := \begin{cases} \mathsf{succ}(f_i = |S| + i) & \text{if } p_i = f_i \\ \mathsf{succ}(f_i' = |S| + i) \sqcap \mathsf{succ}(|r_i \cap (f_i' = \mathtt{next}\, f_i)| \geq 1) & \text{if } p_i = r_i f_i. \end{cases}$$

for $i = 0, 1$, with fresh names $S \in \mathsf{N_C}$, $f_i' \in \mathsf{N_F}$. □

Local and global cardinality constraints. It is possible to extend $\mathcal{ALCSCC}$ by replacing succ-restrictions, ranging over sets of role successors, with sat-*restrictions* $\mathsf{sat}(\mathfrak{con})$ ranging over the whole domain of an interpretation. For the resulting DL, called $\mathcal{ALCSCC}^{++}$, the consistency problem is NExpTime-complete [4]. In this DL, we can state that an individual likes *all* existing cars using the concept $\mathsf{sat}(\mathsf{likes} \cap \mathsf{Car} = \mathsf{Car})$; in contrast, $\mathsf{succ}(\mathsf{likes} \cap \mathsf{Car} = \mathsf{Car})$ describes an individual that likes all cars to which it is related by some role.

If we consider the DL $\mathcal{ALCSCC}^{++}(\mathfrak{D})$ obtained by adding sat-restrictions in the presence of concrete domains, then these restrictions may additionally contain feature roles. For example, the concept $\mathsf{sat}(\top = (\mathsf{age} \geq \mathtt{next}\, \mathsf{age}))$ describes the *overall* oldest individuals, by saying that their age is greater or equal to those of all individuals, while $\mathsf{succ}(\top = (\mathsf{age} \geq \mathtt{next}\, \mathsf{age}))$ describes individuals that are not younger than any individuals related to them by some role name.

Formally, both $\mathcal{ALCSCC}^{++}$ and $\mathcal{ALCSCC}^{++}(\mathfrak{D})$ are evaluated over *finite* interpretations. In [4], this has been used to show that the consistency problem for the extension of $\mathcal{ALCSCC}^{++}$ with inverse roles is undecidable. Similarly, we can use sat-restrictions with feature roles to simulate multiplication of cardinalities of *finite* sets, and thus reduce Hilbert's tenth problem [28] to the consistency of a $\mathcal{ALCSCC}^{++}(\mathfrak{D})$ TBox, provided that $\mathfrak{D}$ is JD. Writing $C \equiv D$ as a shorthand for $C \sqsubseteq D$ and $D \sqsubseteq C$, we can encode the equation $\mathfrak{e} = (x = y \cdot z)$ over integers as a product of cardinalities $|A_x^{\mathcal{I}}| = |A_y^{\mathcal{I}}| \cdot |A_z^{\mathcal{I}}|$, in three steps. First, we enforce $r_{\mathfrak{e}}^{\mathcal{I}} = A_y^{\mathcal{I}} \times A_z^{\mathcal{I}}$ to hold with $A_y \equiv \mathsf{sat}(|r_{\mathfrak{e}}| \geq 1)$ and $A_y \equiv \mathsf{sat}(r_{\mathfrak{e}} = A_z)$; then, we enforce $|s_{\mathfrak{e}}^{\mathcal{I}}| = |A_x^{\mathcal{I}}|$ by adding $\top \sqsubseteq \mathsf{sat}(s_{\mathfrak{e}} = (f_{\mathfrak{e}} = \mathtt{next}\, g_{\mathfrak{e}}))$ and the CIs

$$\top \sqsubseteq \mathsf{sat}(|(g_{\mathfrak{e}} = \mathtt{next}\, f_{\mathfrak{e}})| \leq 1) \text{ and } A_x \sqsubseteq \mathsf{sat}(|(g_{\mathfrak{e}} = \mathtt{next}\, f_{\mathfrak{e}})| \geq 1).$$

Finally, we add $\top \sqsubseteq \mathsf{sat}(|r_{\mathfrak{e}}| = |s_{\mathfrak{e}}|)$, so that, for every finite model $\mathcal{I}$ of all these CIs, $|A_x^{\mathcal{I}}| = |s_{\mathfrak{e}}^{\mathcal{I}}| = |r_{\mathfrak{e}}^{\mathcal{I}}| = |A_y^{\mathcal{I}} \times A_z^{\mathcal{I}}| = |A_y^{\mathcal{I}}| \cdot |A_z^{\mathcal{I}}|$ holds.

Theorem 5. *If the concrete domain $\mathfrak{D}$ is infinite and JD, then the consistency problem for $\mathcal{ALCSCC}^{++}(\mathfrak{D})$ TBoxes is undecidable.*

Transitive roles. Often, we may want a role name to be interpreted as a transitive relation: for instance, $\mathsf{trans}(\mathsf{ancestor})$ in the TBox expresses the fact that the ancestor of an ancestor is also an ancestor. The interaction between number restrictions and transitivity axioms in the presence of role inclusions is known to lead to undecidability [20]. It is possible to regain decidability by disallowing transitive roles within number restrictions, even in the presence of inverse roles [20]. Another restriction that leads to decidability is to replace number restrictions with role functionality axioms; in this case, decidability holds even if one additionally allows nominals and inverse roles [16].

In the DL $\mathcal{SSCC}$ that extends $\mathcal{ALCSCC}$ with transitivity axioms, consistency is undecidable even under all syntactic constraints mentioned above. In particular, we require that numerical constraints contain no transitive roles and no constants other than 0 or 1. By adapting the reduction [20] from the tiling problem, which is known to be undecidable [12], we obtain the following.

Theorem 6. *Consistency in $\mathcal{SSCC}$ is undecidable, even if numerical constraints contain no transitive roles and no constants other than* 0 *or* 1*.*

7 Conclusion

We have presented the very expressive DL $\mathcal{ALCOSCC}(\mathfrak{D})$ that supports concrete domain restrictions and role successor restrictions involving feature values. We have shown that consistency in this logic is ExpTime-complete, the same as for the basic DL $\mathcal{ALC}$. Moreover, we have discussed the consequences of adding assertions, transitive roles, unrestricted semantics, or mixed constraints, most of which make the logic undecidable. While feature roles can already express a restricted form of inverse roles, in the future, we would like to investigate the decidability and complexity of $\mathcal{ALCOISCC}(\mathfrak{D})$ with full inverse roles, for which it is known that they increase the complexity of classical DLs with nominals and number restrictions to NExpTime [30]. Another avenue of research is to implement a reasoner for $\mathcal{ALCOSCC}(\mathfrak{D})$, based on a suitable tableaux algorithm [27] that needs to integrate a QFBAPA solver and a concrete domain reasoner. Currently, reasoners for DLs with non-trivial concrete domains only exist for $\mathcal{ALC}(\mathfrak{D})$ and $\mathcal{EL}(\mathfrak{D})$ with so-called p-admissible concrete domains and without feature paths [2].

Acknowledgments. This work was partially supported by DFG grant 389792660 as part of TRR 248 – CPEC, by the German Federal Ministry of Education and Research (BMBF, SCADS22B) and the Saxon State Ministry for Science, Culture and Tourism (SMWK) by funding the competence center for Big Data and AI "ScaDS.AI Dresden/Leipzig".

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Commun. ACM **26**(11), 832–843 (1983). https://doi.org/10.1145/182.358434
2. Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., Kovtunova, A.: Combining Proofs for Description Logic and Concrete Domain Reasoning. In: Fensel, A., Ozaki, A., Roman, D., Soylu, A. (eds.) Rules and Reasoning - 7th International Joint Conference, RuleML+RR 2023, Proceedings. Lecture Notes in Computer Science, vol. 14244, pp. 54–69. Springer (2023). https://doi.org/10.1007/978-3-031-45072-3_4

3. Baader, F.: A New Description Logic with Set Constraints and Cardinality Constraints on Role Successors. In: Dixon, C., Finger, M. (eds.) Proc. of the 11th Int. Symposium on Frontiers of Combining Systems (FroCoS'17). Lecture Notes in Computer Science, vol. 10483, pp. 43–59. Springer-Verlag, Brasília, Brazil (2017). https://doi.org/10.1007/978-3-319-66167-4_3
4. Baader, F., Bednarczyk, B., Rudolph, S.: Satisfiability and Query Answering in Description Logics with Global and Local Cardinality Constraints. In: ECAI 2020. pp. 616–623. IOS Press (2020). https://doi.org/10.3233/FAIA200146
5. Baader, F., Borgwardt, S., De Bortoli, F., Koopmann, P.: Concrete Domains Meet Expressive Cardinality Restrictions in Description Logics (Extended Version) (2025). https://doi.org/10.48550/arXiv.2505.21103
6. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003). https://doi.org/10.1017/CBO9780511711787
7. Baader, F., De Bortoli, F.: On the Expressive Power of Description Logics with Cardinality Constraints on Finite and Infinite Sets. In: Herzig, A., Popescu, A. (eds.) Frontiers of Combining Systems – 12th International Symposium, FroCoS 2019, Proceedings. Lecture Notes in Computer Science, vol. 11715, pp. 203–219. Springer (2019). https://doi.org/10.1007/978-3-030-29007-8_12
8. Baader, F., Hanschke, P.: A Scheme for Integrating Concrete Domains into Concept Languages. In: Mylopoulos, J., Reiter, R. (eds.) Proceedings of the 12th International Joint Conference on Artificial Intelligence, IJCAI 1991. pp. 452–457. Morgan Kaufmann (1991), http://ijcai.org/Proceedings/91-1/Papers/070.pdf
9. Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic. Cambridge University Press (2017). https://doi.org/10.1017/9781139025355
10. Baader, F., Rydval, J.: Description Logics with Concrete Domains and General Concept Inclusions Revisited. In: Peltier, N., Sofronie-Stokkermans, V. (eds.) Automated Reasoning – 10th International Joint Conference, IJCAR 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12166, pp. 413–431. Springer (2020). https://doi.org/10.1007/978-3-030-51074-9_24
11. Baader, F., Rydval, J.: Using Model Theory to Find Decidable and Tractable Description Logics with Concrete Domains. Journal of Automated Reasoning **66**(3), 357–407 (Aug 2022). https://doi.org/10.1007/s10817-022-09626-2
12. Berger, R.: The undecidability of the domino problem. Memoirs of the American Mathematical Society **66**, 72 (1966), https://mathscinet.ams.org/mathscinet-getitem?mr=216954
13. Borgwardt, S., De Bortoli, F., Koopmann, P.: The Precise Complexity of Reasoning in $\mathcal{ALC}$ with ω-Admissible Concrete Domains. In: Giordano, L., Jung, J.C., Ozaki, A. (eds.) Proceedings of the 37th International Workshop on Description Logics (DL 2024). CEUR Workshop Proceedings, vol. 3739. CEUR-WS.org (2024), https://ceur-ws.org/Vol-3739/paper-1.pdf
14. Carapelle, C., Turhan, A.: Description Logics Reasoning w.r.t. General TBoxes Is Decidable for Concrete Domains with the EHD-Property. In: Kaminka, G.A., Fox, M., Bouquet, P., Hüllermeier, E., Dignum, V., Dignum, F., van Harmelen, F. (eds.) ECAI 2016 – 22nd European Conference on Artificial Intelligence. Frontiers in Artificial Intelligence and Applications, vol. 285, pp. 1440–1448. IOS Press (2016). https://doi.org/10.3233/978-1-61499-672-9-1440
15. Demri, S., Quaas, K.: First Steps Towards Taming Description Logics with Strings. In: Gaggl, S.A., Martinez, M.V., Ortiz, M. (eds.) Logics in Artificial Intelligence

– 18th European Conference, JELIA 2023, Proceedings. Lecture Notes in Computer Science, vol. 14281, pp. 322–337. Springer (2023). https://doi.org/10.1007/978-3-031-43619-2_23
16. Gutiérrez-Basulto, V., Ibáñez-García, Y.A., Jung, J.C.: Number Restrictions on Transitive Roles in Description Logics with Nominals. In: Singh, S., Markovitch, S. (eds.) Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp. 1121–1127. AAAI Press (2017). https://doi.org/10.1609/AAAI.V31I1.10678
17. Hollunder, B., Baader, F.: Qualifying Number Restrictions in Concept Languages. In: Allen, J.F., Fikes, R., Sandewall, E. (eds.) Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). pp. 335–346. Morgan Kaufmann (1991)
18. Hollunder, B., Nutt, W., Schmidt-Schauß, M.: Subsumption Algorithms for Concept Description Languages. In: 9th European Conference on Artificial Intelligence, ECAI 1990. pp. 348–353 (1990)
19. Horrocks, I., Sattler, U.: Ontology Reasoning in the $\mathcal{SHOQ}$(D) Description Logic. In: Nebel, B. (ed.) Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001. pp. 199–204. Morgan Kaufmann (2001)
20. Horrocks, I., Sattler, U., Tobies, S.: Practical Reasoning for Very Expressive Description Logics. Log. J. IGPL **8**(3), 239–263 (2000). https://doi.org/10.1093/JIGPAL/8.3.239
21. Jonsson, P.: Constants and Finite Unary Relations in Qualitative Constraint Reasoning. Artificial Intelligence **257**, 1–23 (Apr 2018). https://doi.org/10.1016/j.artint.2017.12.003
22. Kuncak, V., Rinard, M.: Towards Efficient Satisfiability Checking for Boolean Algebra with Presburger Arithmetic. In: Pfenning, F. (ed.) Automated Deduction – CADE-21. Lecture Notes in Computer Science, vol. 4603, pp. 215–230. Springer Berlin Heidelberg, Berlin, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73595-3_15
23. Labai, N., Ortiz, M., Simkus, M.: An ExpTime Upper Bound for $\mathcal{ALC}$ with Integers. In: Calvanese, D., Erdem, E., Thielscher, M. (eds.) Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020. pp. 614–623 (2020). https://doi.org/10.24963/KR.2020/61
24. Li, S., Liu, W., Wang, S.: Qualitative Constraint Satisfaction Problems: An Extended Framework with Landmarks. Artificial Intelligence **201**, 32–58 (Aug 2013). https://doi.org/10.1016/j.artint.2013.05.006
25. Lutz, C.: The complexity of description logics with concrete domains. Ph.D. thesis, RWTH Aachen University, Germany (2002), http://sylvester.bth.rwth-aachen.de/dissertationen/2002/042/index.htm
26. Lutz, C.: NExpTime-complete description logics with concrete domains. ACM Transactions on Computational Logic (TOCL) **5**(4), 669–705 (2004). https://doi.org/10.1145/1024922.1024925
27. Lutz, C., Miličić, M.: A Tableau Algorithm for Description Logics with Concrete Domains and General TBoxes. Journal of Automated Reasoning **38**(1), 227–259 (Apr 2007). https://doi.org/10.1007/s10817-006-9049-7
28. Matiyasevich, Y.V.: Hilbert's tenth problem. With a foreword by Martin Davis. Cambridge, MA: MIT Press (1993)
29. Randell, D.A., Cui, Z., Cohn, A.G.: A Spatial Logic based on Regions and Connection. In: Nebel, B., Rich, C., Swartout, W.R. (eds.) Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92). pp. 165–176. Morgan Kaufmann (1992)

30. Tobies, S.: The Complexity of Reasoning with Cardinality Restrictions and Nominals in Expressive Description Logics. J. Artif. Intell. Res. **12**, 199–217 (2000). https://doi.org/10.1613/JAIR.705
31. Tobies, S.: PSPACE Reasoning for Graded Modal Logics. J. Log. Comput. **11**(1), 85–106 (2001). https://doi.org/10.1093/LOGCOM/11.1.85