SHACL Satisfiability: What can we learn from DLs?

Anouk Michelle Oudshoorn[1]

[1]*Institute of Logic and Computation, TU Wien, Austria*

Abstract

Since the introduction of the SHACL standard, understanding its computational features and formal foundations has become essential. Some research has focused on the semantics of recursive constraints and the complexity of validation, but the satisfiability of SHACL constraints remains largely unexplored. The most significant previous work in this direction is rather coarse, obtaining very few positive results for finite satisfiability and for fragments with counting. In this paper, we build on description logics to paint a comprehensive and fine-grained boundary for SHACL fragments with a decidable satisfiability problem under the supported semantics, both for unrestricted and finite models.

Keywords

SHACL, satisfiability, finite-model property

1. Introduction

Since the SHACL standard was introduced, the need for a solid understanding of its computational features and formal foundations has been apparent. Several works have leveraged related logic formalisms to give semantics to recursive constraints, obtain complexity bounds, and solve basic tasks including validation [1, 2, 3, 4], but little attention has been devoted to the satisfiability of SHACL constraints. This problem is of major importance in the design and validation of SHACL-based solutions. Nevertheless, we have remarkably few results concerning the decidability and complexity. Indeed, the most notable work in this direction, [5], leaves some very important gaps; it is very coarse. It builds on a tailored fragment of predicate logic to identify decidability and complexity bounds, but the basic logic it considers is already very close to the boundary of what could potentially be decidable in the presence of cardinality constraints. The positive results are mostly limited to formalisms that do not support counting, and more often than not consider unrestricted (that is, potentially infinite) graphs, even though finite graphs are a more relevant setting here.

In this paper, we revisit satisfiability under the supported model semantics. We build on Description Logics (DLs), a well-known family of languages for Knowledge Representation and Reasoning that offers decades of research in the fine-grained study of logical fragments and the effect that the interaction between different shapes of subformulas has on the complexity of reasoning. The close relationship between DLs and SHACL is well-known, and in this paper, we leverage it to paint a much finer boundary of SHACL fragments that have decidable satisfiability problems, both over unrestricted graphs and over graphs with a final domain.

DL 2025: 38th International Workshop on Description Logics, September 3–6, 2025, Opole, Poland

anouk.oudshoorn@tuwien.ac.at (A. M. Oudshoorn)

0009-0006-4638-5948 (A. M. Oudshoorn)

Contributions. We build on the DL literature to pinpoint much tighter complexity bounds than previously known for SHACL. These results are based on the close connection between DL satisfiability and SHACL satisfiability; we revisit this connection and explain how to translate complexity results, both membership and hardness, in both ways. To emphasize this tight bond, we provide a DL inspired naming convention for SHACL fragments.

Moreover, we add to the landscape some lack of finite model property results: we show that $\mathcal{ALC}_S$ plus counting over regular path expressions does not have the finite model property, which also provides an alternative undecidability proof. Furthermore, we show that adding either $\mathsf{eq}(E, r)$ or $\mathsf{disj}(E, r)$ to $\mathcal{ALC}_S$ also breaks the finite model property of $\mathcal{ALC}_S$.

Related Literature. There are two other papers considering satisfiability of (recursive) SHACL: [5] and [6]. Both these works are based on a translation of SHACL into a fragment of first-order logic and transferring complexity results. A tool for testing SHACL satisfiability based on this translation is presented in [7]. Our work differs in considering different fragments by starting from a smaller base logic: the smallest logic considered in these two works corresponds to $\mathcal{ALCIO}_S$ extended with universal roles. Another work considering the close connection between SHACL and DLs for deciding complexity of reasoning problems in SHACL is [8]. In this paper, containment of two shapes in a shapes graph is considered. However, as pointed out in [9], there are some issues with their translation.

2. Preliminaries

Data Graphs and Interpretations. Let N_C, N_R and N_I denote countably infinite, mutually disjoint sets of *concept names* (also known as *class names*), *role names* (or, *property names*), and *individuals* (or, *constants*) respectively. Let $\overline{N}_R := \{p, p^- \mid p \in N_R\}$ denote *roles*.For every $p \in N_R$, let $(p^-)^- = p$. An *atom* (or, *assertion*) is an expression of the form $A(c)$ or $p(c, c')$, for $A \in N_C$, $p \in N_R$ and $\{c, c'\} \subseteq N_I$. An *ABox* (or a *data graph*) $\mathcal{A}$ is a finite set of atoms.

An *interpretation* is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set (called *domain*) and $\cdot^{\mathcal{I}}$ is a function that maps every $A \in N_C$ to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, every $p \in N_R$ to a binary relation $p^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and every individual $c \in N_I$ to an element $c^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Let $(p^-)^{\mathcal{I}} := \{(c', c) \mid (c, c') \in p^{\mathcal{I}}\}$. We call an interpretation $\mathcal{I}$ *finite* whenever $A^{\mathcal{I}}$ and $p^{\mathcal{I}}$, for all $A \in N_C$ and $p \in N_R$, are finite and moreover, $\{A \in N_C \mid A^{\mathcal{I}} \neq \emptyset\} \cup \{p \in N_R \mid p^{\mathcal{I}} \neq \emptyset\}$ is finite. We make the standard name assumption, which means $c^{\mathcal{I}} = c$ for all interpretations $\mathcal{I}$, and all $c \in N_I$. The *canonical interpretation* $\mathcal{I}_{\mathcal{A}}$ of a set of atoms $\mathcal{A}$ is defined by setting $\Delta^{\mathcal{I}_{\mathcal{A}}} = N_I$, $A^{\mathcal{I}_{\mathcal{A}}} = \{c \mid A(c) \in \mathcal{A}\}$ for all $A \in N_C$ and $p^{\mathcal{I}_{\mathcal{A}}} = \{(c, c') \mid p(c, c') \in \mathcal{A}\}$ for all $p \in N_R$.

Description Logic $\mathcal{ALCOIQ}$. An $\mathcal{ALCOIQ}$ concept C is defined in the following way:

$$C ::= c \mid A \mid \top \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \geq_n r.C \mid \forall r.C,$$

where $c \in N_I$, $A \in N_C$ and $r \in \overline{N}_R$. An $\mathcal{ALCOIQ}$ TBox $\mathcal{T}$ is a set of axioms of the form $C \sqsubseteq D$, for C and D $\mathcal{ALCOIQ}$ concepts. An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ whenever for all $C \sqsubseteq D \in \mathcal{T}$ we have $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, where $C^{\mathcal{I}}$ is recursively defined as: $(\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$,

$$\begin{aligned}
\top^{\mathcal{I},S} &= N_I & s^{\mathcal{I},S} &= \{c \in \Delta^{\mathcal{I}} \mid s(c) \in S\} \\
c^{\mathcal{I},S} &= \{c^{\mathcal{I}}\} & (\neg\varphi)^{\mathcal{I},S} &= \Delta^{\mathcal{I}} \setminus (\varphi)^{\mathcal{I},S} \\
A^{\mathcal{I},S} &= A^{\mathcal{I}} & (\varphi \wedge \varphi')^{\mathcal{I},S} &= (\varphi)^{\mathcal{I},S} \cap (\varphi')^{\mathcal{I},S} \\
(\geq_n E.\varphi)^{\mathcal{I},S} &= \{c \in \Delta^{\mathcal{I}} \mid |\{c' \in \Delta^{\mathcal{I}} \mid (c,c') \in E^{\mathcal{I}} c' \in \varphi^{\mathcal{I},S}\}| \geq n\} \\
(\mathsf{eq}(E,r))^{\mathcal{I},S} &= \{c \in \Delta^{\mathcal{I}} \mid \{c' \in \Delta^{\mathcal{I}} \mid (c,c') \in E^{\mathcal{I}}\} = \{c' \in \Delta^{\mathcal{I}} \mid (c,c') \in r^{\mathcal{I}}\}\} \\
(\mathsf{disj}(E,r))^{\mathcal{I},S} &= \{c \in \Delta^{\mathcal{I}} \mid \{c' \in \Delta^{\mathcal{I}} \mid (c,c') \in E^{\mathcal{I}}\} \cup \{c' \in \Delta^{\mathcal{I}} \mid (c,c') \in r^{\mathcal{I}}\} = \emptyset\} \\
(\mathsf{closed}(R))^{\mathcal{I},S} &= \{c \in \Delta^{\mathcal{I}} \mid \{r \in \overline{N}_R \setminus R \mid (c,c') \in r^{\mathcal{I}}\} = \emptyset\}
\end{aligned}$$

Figure 1: Evaluating shape expressions

$(C \sqcap C')^{\mathcal{I}} := C^{\mathcal{I}} \cap C'^{\mathcal{I}}$, $(C \sqcup C')^{\mathcal{I}} := C^{\mathcal{I}} \cup C'^{\mathcal{I}}$, $(\geq_n r.C)^{\mathcal{I}} := \{c \in \Delta^{\mathcal{I}} \mid |\{c' \in \Delta^{\mathcal{I}} \mid (c,c') \in r^{\mathcal{I}}, c' \in C^{\mathcal{I}}\}| \geq n\}$ and $(\forall r.C)^{\mathcal{I}} := \{c \in \Delta^{\mathcal{I}} \mid (c,c') \in r^{\mathcal{I}} \rightarrow c' \in C^{\mathcal{I}}\}$.

A concept C is *satisfiable* w.r.t. a TBox $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$.

Recursive Shape Constraint Language (SHACL). We define *shape expressions*, following [10], but adding recursion, in the following way

$$\varphi ::= s \mid c \mid A \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \geq_n E.\varphi \mid \mathsf{eq}(E,r) \mid \mathsf{disj}(E,r) \mid \mathsf{closed}(R),$$

where $c \in N_I$, $A \in N_C$, $n \geq 1$, R a finite subset of $\overline{N}_R$ and E a regular expression given by

$$E ::= r \mid E^* \mid E \circ E \mid E \cup E,$$

for $r \in \overline{N}_R$. Here, $(E^*)^{\mathcal{I}}$ corresponds to the transitive closure of $E^{\mathcal{I}}$, $(E \circ E')^{\mathcal{I}} := \{(c,c') \mid (c,d) \in E^{\mathcal{I}}, (d,c') \in E'^{\mathcal{I}}\}$, and $(E \cup E')^{\mathcal{I}} := E^{\mathcal{I}} \cup E'^{\mathcal{I}}$. We use EE' as a shorthand for $E \circ E'$, and E^+ as a shorthand for EE^*. We set $\varphi \vee \varphi' := \neg(\neg\varphi \wedge \neg\varphi')$ and $\forall E.\varphi := \neg \geq_1 E.\neg\varphi$. A *shape constraint* is an expression of the form $s \leftarrow \varphi$, for $s \in N_S$ and φ a shape expression. Here, s is the *head* of the constraint. We assume each shape name s only appears as the head of one constraint - this does not influence expressibility as '$\vee$' may be used in φ. The semantics of recursive SHACL is defined via a *shape assignment*: a set of shape atoms S, such that the rules in Figure 1 are satisfied, and such that $s \leftarrow \varphi \in \mathcal{C}$ implies $s^{\mathcal{I},S} = (\varphi)^{\mathcal{I},S}$.

Given an interpretation $\mathcal{I}$ and a shape assignment S, we say a node $c \in N_I$ *validates* a shape expression φ, whenever $c \in (\varphi)^{\mathcal{I},S}$, where $(\varphi)^{\mathcal{I},S}$ is inductively defined in Table 1. Let $\mathcal{G}$ be a set of targets of the form $s(c)$, which we call *atomic targets*, or $s(A)$, for s a shape name, $c \in N_I$ and $A \in N_C$. A pair $(\mathcal{C},\mathcal{G})$ is called a *shapes graph*. In this paper, we consider the *supported model semantics*, that is, given an interpretation $\mathcal{I}$, we say $\mathcal{I}$ *validates* $(\mathcal{C},\mathcal{G})$ when there exists a shape assignment S of the constraints in $\mathcal{C}$ such that for all $s(c) \in \mathcal{G}$, we find c validates s, and for all $s(A) \in \mathcal{G}$, all nodes in $\mathcal{A}^{\mathcal{I}}$ validate s. Considering readability, we will write $\mathcal{A}$ validates $(\mathcal{C},\mathcal{G})$, for any set of atoms $\mathcal{A}$, in which case the canonical interpretation $\mathcal{I}_{\mathcal{A}}$ is intended.

3. SHACL Satisfiability

In this paper we study the following reasoning problems:

Satisfiability: Given a shapes graph $(\mathcal{C},\mathcal{G})$, decide whether there exists an interpretation $\mathcal{I}$ that validates $(\mathcal{C},\mathcal{G})$.

Finite Satisfiability: Given a shapes graph $(\mathcal{C}, \mathcal{G})$, decide whether there exists an interpretation $\mathcal{I}$ with a finite number of nodes, that is, $\Delta^{\mathcal{I}}$ is finite, that validates $(\mathcal{C}, \mathcal{G})$.

Finite Model Property: A SHACL fragment has the *finite model property* iff for every shapes graph $(\mathcal{C}, \mathcal{G})$ expressible in the given fragment, we find that if $(\mathcal{C}, \mathcal{G})$ is satisfiable, then $(\mathcal{C}, \mathcal{G})$ is finitely satisfiable.

Clearly, having the finite model property extends to less expressive fragments, whereas the opposite, not having the property, spreads to subsuming fragments. Similarly, for (finite) satisfiability, membership of a complexity class spreads to less expressive fragments, and hardness to the more expressive ones. In case a fragment has the finite model property, the membership results for general satisfiability extend to the finite setting.

The above presented problems are not the only ones one might consider: in [6], two flavours of the SHACL satisfiability problem are discussed. The first one is satisfiability as described above, whereas the second option is referred to as *constraint satisfiability*: the satisfiability problem when the set of constraints only consists of a single constraint. As already noted in [6], the constraint version of the problem clearly reduces to the general version, which means upper bounds for complexity are preserved. We show here that for recursive SHACL also the other reduction, constraint to general satisfiability, holds. But first, we note that for satisfiability purposes, we may restrict the form of the targets.

Lemma 1. *For each shapes graph $(\mathcal{C}, \mathcal{G})$ there exists a shapes graph $(\mathcal{C}', \mathcal{G}')$ such that $\mathcal{G}'$ only exists of atomic targets and for each model $\mathcal{I}$ we have $\mathcal{I}$ validates $(\mathcal{C}, \mathcal{G})$ iff $\mathcal{I}$ validates $(\mathcal{C}', \mathcal{G}')$.*

Proof. To see this, assume that for some concept name $A \in N_C$, we find the target $s(A) \in \mathcal{G}$. It is easy to check it suffices to replace each occurrence of A in $\mathcal{C}$ by $(A \wedge s)$, and remove all $s(A)$ from $\mathcal{G}$, and do the same for any other $s'(B) \in \mathcal{G}$. □

Proposition 1. *In recursive SHACL, the problems of deciding SHACL satisfiability and constraint satisfiability coincide.*

Proof. We use '$\varphi \leftrightarrow \psi$' as a shorthand for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$, and '$\varphi \rightarrow \psi$' for $\neg\varphi \vee \psi$. Given a shapes graph $(\mathcal{C}, \mathcal{G})$, such that all targets in $\mathcal{G}$ are atomic. We distinguish two cases.

In case the SHACL fragment does not contain nominals, satisfiability of $(\mathcal{C}, \mathcal{G})$ is equivalent to satisfiability of $(\mathcal{C}, \mathcal{G}_c)$, where $\mathcal{G}_c := \{s(c) \in \mathcal{G}\}$, for all $c \in N_I$ such that c appears in $\mathcal{G}$. Furthermore, note we may replace multiple targets using the same c by a single target $s(c)$ for some fresh shape name s, given we add $s \leftarrow \bigwedge_{s'(c) \in \mathcal{G}} s'$ to the set of constraints. Thus, we may assume without loss of generality that $\mathcal{G} = \{s(c)\}$ when testing satisfiability of $(\mathcal{C}, \mathcal{G})$.

Note that we may also encode all constraints within a single one: satisfiability of $(\mathcal{C}, \{s(c)\})$ can be reduced to satisfiability of $(\{\hat{s} \leftarrow \hat{\varphi}\}, \{\hat{s}(c)\})$, for a fresh shape name $\hat{s}$, and $\hat{\varphi}$ defined in the following way:

$$\hat{\varphi} := s \wedge \forall(\prod_{r \in R} r)^*. \bigwedge_{s' \leftarrow \varphi \in \mathcal{C}} (s' \leftrightarrow \varphi),$$

where $R \subseteq \overline{N}_R$ contains all roles appearing in any constraint in $\mathcal{C}$.

Name	Syntax	Symbol
Nominals	c	$\mathcal{O}$
Inverses	r^-	$\mathcal{I}$
Functionality	$\leq_1 r.\top$	$\mathcal{F}$
Unqualified number restriction	$\geq_n r.\top$	$\mathcal{N}$
Qualified number restriction	$\geq_n r.\varphi$	$\mathcal{Q}$
Unqualified regular path counting	$\geq_n E.\top$	$\mathcal{E}$
Qualified regular path counting	$\geq_n E.\varphi$	$\mathcal{P}$

Table 1
Fragments of SHACL following the DL naming convention, extended with counting over regular paths.

In the case the SHACL fragment does contain nominals, the above described reduction to single-element targets may no longer be sound. Instead, we use the nominals in the newly defined constraint in the following way: satisfiability of $(\mathcal{C}, \mathcal{G})$ may be reduced to satisfiability of $(\{\hat{s} \leftarrow \hat{\varphi}\}, \{\hat{s}(c)\})$ for each $c \in N_I$ appearing in $\mathcal{G}$, such that

$$\hat{\varphi} := \forall(\bigsqcap_{r \in R} r)^*. \bigwedge_{s(c) \in \mathcal{G}} (c \to s) \wedge \bigwedge_{s \leftarrow \varphi \in \mathcal{C}} (s \leftrightarrow \varphi),$$

where $R \subseteq \overline{N}_R$ contains all roles appearing in any constraint in $\mathcal{C}$. □

Names for fragments of SHACL. Let $\mathcal{ALC}_S$ be the fragment of SHACL such that shape expressions φ are of the form:

$$\varphi ::= s \mid A \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \exists r.\varphi \mid \forall r.\varphi,$$

for $r \in N_R$. Let $\exists r.\varphi$ be a shorthand for $\geq_1 r.\varphi$. Furthermore, we set $\forall r.\varphi := \neg\exists r.\neg\varphi$. Partly following the naming convention of Description Logics, we name the SHACL fragments in the way presented in Table 1. We write $\mathcal{L}X_S$ to denote the SHACL fragment by extending $\mathcal{L}_S$ with the features described by some $X \in \{\mathcal{O}, \mathcal{I}, \mathcal{F}, \mathcal{N}, \mathcal{Q}, \mathcal{E}, \mathcal{P}\}$. With the superscript $\mathcal{L}^e$, we denote that the feature $\mathsf{eq}(r, r')$, for $\{r, r'\} \subseteq N_R$ is added to the fragment $\mathcal{L}$. Similarly, $\mathcal{L}^d$ corresponds to adding the feature $\mathsf{disj}(r, r')$, also for $\{r, r'\} \subseteq N_R$.

Note that constraints containing $\mathsf{closed}(R)$ already fall into the fragment $\mathcal{ALC}_S$. The same can not be said about $\geq_1 E.\varphi$: although this is reducible to $\mathcal{ALC}_S$ under the least-fixed point semantics [11], this does not extend to the supported model semantics.

Lemma 2. *Every constraint of the form* $\mathsf{closed}(R)$ *may be reduced to* $\mathcal{ALC}_S$*, when considering satisfiability problems.*

Proof. Since we are in the restricted context of SHACL satisfiability, that is, roles not mentioned in the constraints are irrelevant, we may reduce '$\mathsf{closed}(R)$' to '$\bigsqcap_{r \in R^c} \neg\exists r.\top$', where $R^c := \{r \in N_R \setminus R \mid r \text{ appears in } \mathcal{C}\}$. □

4. SHACL to OWL and back again

Most of the results in this work are based on the tight connection between SHACL and DLs. In this section, we look at their close connection and provide a translation for satisfiability

purposes. First we address different forms of recursion, followed by a concrete translation. Finally, this translation is used to say something regarding the joint satisfiability of SHACL and OWL.

Two types of recursion in SHACL. When translating axioms of the form $\top \equiv C$ to SHACL, there are in general three different solutions possible, dependent on the type of recursion allowed in the considered SHACL fragment. Note that we are considering satisfiability, which means we may assume a model exists iff a connected model exists.

Let for $\{r_0, \ldots, r_n\} \subseteq \overline{N}_R$ be all roles appearing in any axiom in a given TBox, that is, both p and p^- whenever both are mentioned separately. Now the first option to translate $\top \equiv C$ is to rely on one of the two forms of recursion SHACL contains. The first version is the recursion intended when talking about the difference between recursive and non-recursive SHACL; the possibility to create cycles of shape names referring to each other, that is, we propagate C by using $s \leftarrow \forall(r_0 \cup \ldots \cup r_n).(s \wedge C)$, whereas the second one is the more direct approach that uses '$*$' and '$\cup$' by setting $s \leftarrow \forall(r_0 \cup \ldots \cup r_n)^*.C$. Having these tricks in mind, we assume no axioms of the form $\top \equiv C$ appear in any TBox in the rest of this section.

Another option is to propagate a specifically chosen concept name A to every node of the graph we are validating. This can be enforced by replacing each $\exists r.\varphi$, for any shape expression φ, by $\exists r.\varphi \wedge \forall r.A$. Finally, taking this A to be the target class of a freshly created constraint $s_D \leftarrow D$ suffices to imply that all relevant nodes of the graph satisfy D.

Translation. We note that for many DLs, it is easy to check that the decidability results known for that logic work also when replacing all '$\sqsubseteq$' by '$\equiv$' in the given constructions, and/or it is well-known that the restriction to axioms only using equivalence does not influence the expressivity of the given logic. In these cases, we may also assume without loss of generality, that one side of the equivalence is a concept name. That is, in this paper, it will be sufficient to consider axioms of the form $A \equiv C$. As we set $s \leftarrow \varphi \in \mathcal{C}$ implies $(s)^{\mathcal{I},S} = (\varphi)^{\mathcal{I},S}$, this aligns well with the semantics of recursive SHACL we are considering.

Let us define two translations: f, a function translating any $\mathcal{ALCOIQ}$ concept into a shape expression, and g, a function in the opposite direction, translating any shape expression expressible in $\mathcal{ALCOIQ}_S$ into an $\mathcal{ALCOIQ}$ concept. These functions are recursively defined in Table 2, where $s_A \in N_S$ is a fresh shape name introduced for every concept name $A \in N_C$, and A_s is a fresh concept name for each $s \in N_S$.

Proposition 2. *Let $\mathcal{T}$ be an $\mathcal{ALCOIQ}$ TBox such that all axioms are of the form $A \equiv C$. Then, $\mathcal{I}$ is a model of $\mathcal{T}$ such that $A^{\mathcal{I}} \neq \emptyset$, iff $\mathcal{I}$ validates $(\mathcal{C}, \mathcal{G})$ given by $\mathcal{G} = \{s_A(c) \mid c \in A^{\mathcal{I}}\}$ and*

$$\mathcal{C} = \{s_A \leftarrow f(C) \wedge A \mid A \equiv C \in \mathcal{T}\} \cup \{s_A \leftarrow A \mid A \equiv C \notin \mathcal{T}\}.$$

Proposition 3. *Let $(\mathcal{C}, \mathcal{G})$ be any $\mathcal{ALCOIQ}_S$ shapes graph, such that $\mathcal{G}$ is atomic. Then $\mathcal{I}$ validates $(\mathcal{C}, \mathcal{G})$, based on the shape assignment S, iff $\mathcal{I}'$, given by $\Delta^{\mathcal{I}} = \Delta^{\mathcal{I}'}$, for all $A \in N_C \setminus \{A_s \mid s \in N_S\}$, $A^{\mathcal{I}} = A^{\mathcal{I}'}$ and for all $A \in \{A_s \mid s \in N_S\}$, $A_s^{\mathcal{I}'} = \{c \in N_I \mid s(c) \in S\}$ is a model of $\{A_s \equiv g(\varphi) \mid s \leftarrow \varphi \in \mathcal{C}\}$, such that if $s(c) \in \mathcal{G}$, then $c \in A_s^{\mathcal{I}'}$.*

Note that correctness of both propositions is based on the fact that shape and concept names can take over each others roles.

$$
\begin{array}{rcl}
f(c) := c & \qquad & g(c) := c \\
f(A) := s_A & & g(A) := A,\, g(s) := A_s \\
f(\top) := \top & & g(\top) := \top \\
f(\neg C) := \neg f(C) & & g(\neg\varphi) := \neg g(\varphi) \\
f(C \sqcap C') := f(C) \wedge f(C') & & g(\varphi \wedge \varphi') := g(\varphi) \sqcap g(\varphi') \\
f(C \sqcup C') := f(C) \vee f(C') & & g(\varphi \vee \varphi') := g(\varphi) \sqcup g(\varphi') \\
f(\geq_n r.C) := \geq_n r.f(C) & & g(\geq_n r.\varphi) := \geq_n r.g(\varphi) \\
f(\forall r.C) := \forall r.f(C) & & g(\forall r.\varphi) := \forall r.g(\varphi)
\end{array}
$$

Figure 2: Translation functions mapping $\mathcal{ALCOIQ}$ concepts into $\mathcal{ALCOIQ}_S$ shape expressions, and vice versa.

Joint Satisfiability of SHACL and OWL As envisioned in the W3C SHACL specification [12, Section 1.5] and argued in [3], it is promising to combine SHACL and OWL, a set of ontology languages, based on DLs. In fact, it is shown in [3] and [11] that validation of SHACL in presence of ontologies can be reduced to plain SHACL validation. Also the complexity of validation is discussed there. However, nothing is known regarding joint satisfiability of SHACL and OWL, that is, the following reasoning problems.

Joint Satisfiability: Given a shapes graph $(\mathcal{C}, \mathcal{G})$ and a TBox $\mathcal{T}$, decide whether there exists an interpretation $\mathcal{I}$ that validates $(\mathcal{C}, \mathcal{G})$ and is a model of $\mathcal{T}$.

Finite Joint Satisfiability: Given a shapes graph $(\mathcal{C}, \mathcal{G})$ and a TBox $\mathcal{T}$, decide whether there exists an interpretation $\mathcal{I}$ with a finite number or nodes, that is $\Delta^{\mathcal{I}}$ is finite, that validates $(\mathcal{C}, \mathcal{G})$ and is a model of $\mathcal{T}$.

Given the above translation, for satisfiability purposes, it follows that the complexity of deciding (finite) Joint Satisfiability of SHACL in presence of OWL corresponds to the complexity of deciding (finite) satisfiability in the least-expressive description logic capturing the expressivity of both the translated SHACL fragment, as the OWL fragment.

5. Inverses, Nominals and Counting

The following propositions are well-known results in the Description Logic community. Combined with a translation as described in the previous section, which naturally extends to the case where roles are replaced by regular expressions, these results extend to our setting.

Proposition 4 (for instance [13, 14, 15])**.** *$\mathcal{ALCIO}_S$ and $\mathcal{ALCOQ}_S$ have the finite model property, $\mathcal{ALCIF}_S$ does not.*

Proposition 5 ([15, 16], and their references)**.** *Deciding (finite) satisfiability in $\mathcal{ALC}_S$, $\mathcal{ALCIO}_S$, $\mathcal{ALCOQ}_S$, and $\mathcal{ALCIQ}_S$ is ExpTime-complete.*

Proposition 6. *Deciding (finite) satisfiability in $\mathcal{ALCIOF}_S$ and $\mathcal{ALCIOQ}_S$ is NExpTime-complete.*

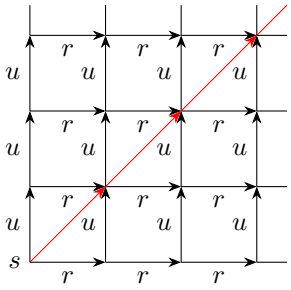

Figure 3: Infinite grid that, after adding s_f and s_g as label to every node, shows satisfiability of s. The red diagonal arrows denote the role d.

The lower bound for $\mathcal{ALCIOF}_S$ follows from constructing a torus of finite size [17]; the upper bound from translating $\mathcal{ALCOIQ}_S$ into the two-variable fragment of first-order logic with counting quantifiers C^2 [18], in which (finite) satisfiability is NExpTime-complete [19].

Proposition 7. *$\mathcal{ALCE}_S$ and $\mathcal{ALCP}_S$ do not have the finite model property.*

Proof. Consider the following constraints, with the target $s(0,0)$:

$$\begin{aligned} s &\leftarrow \forall u^+.\, {=_1}\, r^+ d.\top \wedge \forall (r \cup u)^*.(s_f \wedge s_g) \\ s_f &\leftarrow {=_1}\, u.\top \wedge {=_1}\, r.\top \wedge {=_1}\, (ru \cup ur).\top \\ s_g &\leftarrow \neg {\geq_1}\, d.\top \vee \forall u^+.\neg {\geq_1}\, d.\top \end{aligned}$$

Here, ${=_1}\,.\varphi$ is a shorthand for ${\geq_1}\,.\varphi \wedge {\leq_1}\,.\varphi$. Clearly, a way to satisfy the above constraints is in a simple grid on the natural numbers with a diagonal, where s true in $(0,0)$ and s_f and s_g validated everywhere. Here the interpretation of d is $\{(i,i),(i+1,i+1) \mid i \in \mathbb{N}\}$, for u it is $\{((i,j),(i,j+1)) \mid \{i,j\} \subseteq \mathbb{N}\}$, and for r the set $\{((i,j),(i+1,j)) \mid \{i,j\} \subseteq \mathbb{N}\}$.

Assume for contradiction there exists a finite model. As s_f must hold in s and every node reachable by u, there exists $a_0, \ldots, a_i$ such that a_0 is reachable by u^* from $(0,0)$ and $\{(a_0,a_1),\ldots,(a_{i-1},a_i),(a_i,a_0)\}$ is contained in the interpretation of u. Note that because of having to validate ${=_1}\, r.\top \wedge {=_1}\, (ru \cup ur).\top$ in every node reachable by r or u, it can be concluded that the set of nodes $\{b_0,\ldots,b_j\}$ reachable by r from any node in $\{a_0,\ldots,a_i\}$ must also contain a loop in the interpretation of u. Clearly, this generalises to: every node reachable by r^+ from any node in $\{a_0,\ldots,a_i\}$ has a u^+-path leading to itself. As every node appearing in a loop of u's cannot have an outgoing d-edge, because of the constraint $s_g \leftarrow \neg {\geq_1}\, d.\top \vee \forall u^+.\neg {\geq_1}\, d.\top$, it follows that every node reachable by r^+ from any node in $\{a_0,\ldots,a_i\}$ cannot have an outgoing d-edge. As all nodes in $\{a_0,\ldots,a_i\}$ are reachable by u^+ from $(0,0)$, we cannot validate the first conjunct of s in $(0,0)$. This is the contradiction which concludes the proof. □

The above proof produces a grid, thus it is possible to reduce the undecidable domino problem [20] to $\mathcal{ALCE}_S$, and its subsuming SHACL fragments, making the satisfiability problem undecidable. Note that this undecidability result is already known for different sublogics of $\mathcal{ALCE}_S$. This is discussed in the remainder of this section.

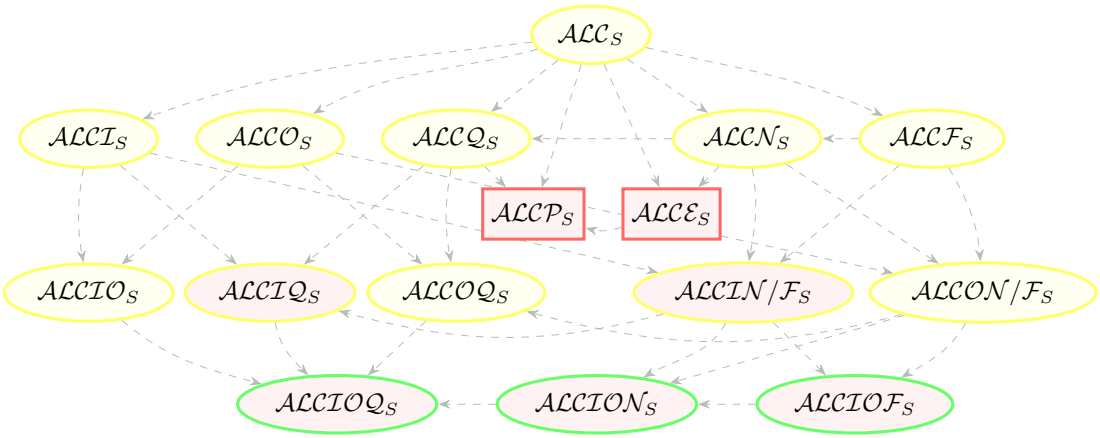

Figure 4: Decidability and complexity of SHACL fragments. In ellipse nodes, deciding (finite) satisfiability is decidable in ExpTime (yellow border), or NExpTime (green border). In squared nodes, deciding satisfiability is undecidable. A yellow filling indicates the presence of the finite model property, whereas a red filling denotes the lack of it. Arrows indicate subsumption of fragments.

More Fine-Grained Analysis. In the following, we will restrict the expressivity of the regular expressions used in $\geq_n E.\top$ and $\geq_n E.\varphi$. That is, with $\mathcal{ALCN}(X)_S$ or $\mathcal{ALCQ}(X)_S$, for X any combination of the role constructs $*, \circ$ and $\cup$, we denote the SHACL fragment allowing regular expressions build from only the role constructs in X in number restrictions. That is, $\mathcal{ALCN}(*, \circ, \cup)_S = \mathcal{ALCE}_S$ and $\mathcal{ALCQ}(*, \circ, \cup)_S = \mathcal{ALCP}_S$. We note that the translation presented in Section 4 naturally extends to also capture $*$, $\circ$ and $\cup$ in the number restrictions.

Again, we can rely on the vast DL literature: the derived complexity results are the following.

Proposition 8. *Deciding satisfiability in $\mathcal{ALCN}(\circ)_S$ is undecidable.*

This is a direct consequence of Theorem 6 in [21].

Proposition 9. *Deciding satisfiability in $\mathcal{ALCN}(*, \cup)_S$ is undecidable.*

Proof. We may adapt the undecidability proof of unrestricted $\mathcal{SHN}$ in [22] in the following way. That is, instead of using the hierarchy and the given axioms, we use the following.

$$s_A \leftarrow \neg s_B \wedge \neg s_C \wedge \neg s_D \wedge \exists x_1.s_B \wedge \exists y_1.s_C \leq_3 (x_1 \cup y_1)^*.\top$$
$$s_B \leftarrow \neg s_A \wedge \neg s_C \wedge \neg s_D \wedge \exists x_2.s_A \wedge \exists y_1.s_D \leq_3 (x_2 \cup y_1)^*.\top$$
$$s_C \leftarrow \neg s_A \wedge \neg s_B \wedge \neg s_D \wedge \exists x_1.s_D \wedge \exists y_2.s_A \leq_3 (x_1 \cup y_2)^*.\top$$
$$s_D \leftarrow \neg s_A \wedge \neg s_B \wedge \neg s_C \wedge \exists x_2.s_C \wedge \exists y_2.s_B \leq_3 (x_2 \cup y_2)^*.\top$$

Now satisfiability of $s_A(c)$ corresponds to existence of a grid, for which we may also write constraints encoding a domino tiling problem, like in [23]. Thus, the undecidability of the domino problem transfers to this logic, which concludes our proof. □

Deciding (finite) satisfiability in $\mathcal{ALCQ}(\cup)_S$ is ExpTime-complete. This result is subsumed by Proposition 12 in the next section.

6. Equality and Disjointness

Recall we introduced the superscripts $\mathcal{L}^d$ and $\mathcal{L}^e$ to denote the addition of the features $\mathsf{disj}(r, r')$ and $\mathsf{eq}(r, r')$, respectively. Following the naming convention introduced in the previous section, for X any combination of the role constructs $*$, $\circ$ and $\cup$, we denote by $\mathcal{L}(X)^d$, resp. $\mathcal{L}(X)^e$, the SHACL fragment allowing one regular expressions build from only the role constructs in X in the disjointness, resp. equality features. That is, recursive SHACL as introduced in the preliminaries and for satisfiability purposes, corresponds to $\mathcal{ALCOIQ}(*, \circ, \cup)^{d,e}_S$.

First, we consider the positive part: adding disjointness does not increase complexity, although the finite model property is easily lost.

Proposition 10. *Deciding satisfiability in $\mathcal{ALCI}(*, \circ, \cup)^{\mathbf{d}}_S$ is ExpTime-complete, and this fragment does not have the finite model property. In fact, $\mathcal{ALC}(*, \circ)^{\mathbf{d}}_S$ already lacks the finite model property.*

Proof. The upper bound follows from Theorem 4.8 in [24]. To see this, note that $\mathsf{disj}(E, r)$ is equivalent to the expression $\forall(E \cap r).\bot$. As the amount of nestings of '$\cap$' in this expression is bounded by a constant, namely 1, the tighter upper bound of ExpTime can be derived.

For the lack of finite model property, consider the following shapes graph $(\mathcal{C}, \mathcal{G})$:

$$\mathcal{C} = \{s \leftarrow \mathsf{disj}(rr^+, r) \wedge \exists r.s\}$$

and set $\mathcal{G} = \{s(a)\}$. Clearly, the infinite chain of r's, in which every node is labelled with an s is an infinite model. In fact, it must be possible to homomorphically map this chain into any interpretation that validates $(\mathcal{C}, \mathcal{G})$. As $\mathsf{disj}(rr^+, r)$ has to be true in each node on the chain, it suffices to check that each approach to loop this chain breaks this disjointness. □

Even though equality and disjointness might appear to be duals, this belief is quickly crashed: equality is much harder and easily leads to undecidability.

Proposition 11. *Deciding satisfiability in $\mathcal{ALC}(\circ)^{\mathbf{e}}_S$ is undecidable and $\mathcal{ALC}(*, \circ)^{\mathbf{e}}_S$ does not have the finite model property.*

Proof. The undecidability result directly follows from results for Description Logics with role value maps [25]. An easy way to also see why the equality feature leads to undecidability is the following constraint set, which encodes a grid.

$$s \leftarrow \mathsf{eq}(ur, d) \wedge \mathsf{eq}(ru, d) \wedge \exists r.s \wedge \exists u.s \wedge \forall r.s \wedge \forall u.s$$

For the lack of finite model property, consider the following shapes graph $(\mathcal{C}, \mathcal{G})$:

$$\mathcal{C} = \{s \leftarrow \mathsf{eq}(r^*, t) \wedge \neg\mathsf{eq}(r^+, t) \wedge \exists r.s\},$$

and set $\mathcal{G} = \{s(a)\}$. Clearly, the infinite chain of r's, with t the transitive closure of r, in which every node is labelled with an s is an infinite model. In fact, it must be possible to homomorphically map this chain into any interpretation that validates $(\mathcal{C}, \mathcal{G})$. As $\mathsf{disj}(rr^+, r)$ has to be true in each node on the chain, it suffices to check that each approach to loop this chain breaks this disjointness. □

It looks much better when solely allowing '$\cup$' in the equality and disjointness axioms: (finite) satisfiability in $\mathcal{ALC}(\cup)_S^{\mathbf{d},\mathbf{e}}$ is ExpTime-complete. In fact, this holds for much stronger fragments.

Proposition 12. *Deciding satisfiability in $\mathcal{ALCIQ}(\cup)_S^{\mathbf{d},\mathbf{e}}$, and (finite) satisfiability in $\mathcal{ALCOQ}(\cup)_S^{\mathbf{d},\mathbf{e}}$ and $\mathcal{ALCOI}(\cup)_S^{\mathbf{d},\mathbf{e}}$ is ExpTime-complete, and the latter two fragments have the finite model property.*

Proof. Note that for R a union of roles, $\mathsf{eq}(R, r)$ may be reduced to $\forall((R \setminus r) \cup (r \setminus R)).\bot$, where $E \setminus E' := E^{\mathcal{I}} \setminus E'^{\mathcal{I}}$, and $\mathsf{disj}(R, r)$ to $\forall(R \cap r).\bot$. Thus, in case only '$\cup$' is allowed, the equality and disjointness features reduce to simple roles, which means the above fragments can be reduced to the description logics $\mathcal{ZIQ}$, $\mathcal{ZOQ}$, resp. $\mathcal{ZOI}$. For all these logics, satisfiability is known to be decidable in ExpTime [26]. Furthermore, $\mathcal{ZOQ}$, and $\mathcal{ZOI}$ have the finite model property [27]. □

Proposition 13. *Deciding (finite) satisfiability in $\mathcal{ALCI}(*)_S^{\mathbf{e}}$ is ExpTime-complete, and this fragment has the finite model property.*

Proof. This fragment reduces to the description logic $\mathcal{SHI}$, of which the above listed properties are well-known, see for instance [18]. □

We note that the results described in this paper do not provide a complete picture of all known decidability results in the DL setting.

7. Conclusion and Outlook

We looked at the tight connection between Description Logics and SHACL. In this way, we derived many new complexity results for deciding (finite) satisfiability in SHACL. Specifically, for the general satisfiability problem the picture looks quite complete: as far as the author knows, only some small fragments remain unclear, like $\mathcal{ALC}(*, \cup)_S^{\mathbf{e}}$, or $\mathcal{ALC}(*)_S^{\mathbf{d},\mathbf{e}}$. However, when looking at finite satisfiability, the status is quite the opposite. Specifically in the setting of SHACL, one of the standard tools for managing concrete data sets, the latter case is of uttermost importance. Thus, a lot of work remains to be done.

Another approach would be to look at different semantics: in this paper, we considered (finite) satisfiability under the supported model semantics. There are different reasons for considering certain semantics for recursive SHACL, and the supported model semantics is definitely not always the preferred option. For future work it will also be relevant to consider reasoning problems for any of the other semantics, for instance with better complexity bounds for deciding validation. To really have a good understanding of what would be optimised SHACL-based solutions, also understanding the complexity of deciding containment under different semantics would be relevant.

References

[1] J. Corman, J. L. Reutter, O. Savkovic, Semantics and validation of recursive SHACL, in: D. Vrandecic, K. Bontcheva, M. C. Suárez-Figueroa, V. Presutti, I. Celino, M. Sabou,

L. Kaffee, E. Simperl (Eds.), The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I, volume 11136 of *Lecture Notes in Computer Science*, Springer, 2018, pp. 318–336. URL: https://doi.org/10.1007/978-3-030-00671-6_19. doi:10.1007/978-3-030-00671-6_19.

[2] M. Andresel, J. Corman, M. Ortiz, J. L. Reutter, O. Savkovic, M. Simkus, Stable model semantics for recursive SHACL, in: Y. Huang, I. King, T. Liu, M. van Steen (Eds.), WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020, ACM / IW3C2, 2020, pp. 1570–1580. URL: https://doi.org/10.1145/3366423.3380229. doi:10.1145/3366423.3380229.

[3] S. Ahmetaj, M. Ortiz, A. Oudshoorn, M. Simkus, Reconciling SHACL and ontologies: Semantics and validation via rewriting, in: K. Gal, A. Nowé, G. J. Nalepa, R. Fairstein, R. Radulescu (Eds.), ECAI 2023 - 26th European Conference on Artificial Intelligence, volume 372 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2023, pp. 27–35. URL: https://doi.org/10.3233/FAIA230250. doi:10.3233/FAIA230250.

[4] C. Okulmus, M. Simkus, SHACL validation under the well-founded semantics, in: P. Marquis, M. Ortiz, M. Pagnucco (Eds.), Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024, 2024. URL: https://doi.org/10.24963/kr.2024/52. doi:10.24963/KR.2024/52.

[5] P. Pareti, G. Konstantinidis, F. Mogavero, Satisfiability and containment of recursive SHACL, J. Web Semant. 74 (2022) 100721. URL: https://doi.org/10.1016/j.websem.2022.100721. doi:10.1016/J.WEBSEM.2022.100721.

[6] P. Pareti, G. Konstantinidis, F. Mogavero, T. J. Norman, SHACL satisfiability and containment, in: J. Z. Pan, V. Tamma, C. d'Amato, K. Janowicz, B. Fu, A. Polleres, O. Seneviratne, L. Kagal (Eds.), The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I, volume 12506 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 474–493. URL: https://doi.org/10.1007/978-3-030-62419-4_27. doi:10.1007/978-3-030-62419-4_27.

[7] P. Pareti, SHACL2FOL: an FOL toolkit for SHACL decision problems, CoRR abs/2406.08018 (2024). URL: https://doi.org/10.48550/arXiv.2406.08018. doi:10.48550/ARXIV.2406.08018. arXiv:2406.08018.

[8] M. Leinberger, P. Seifer, T. Rienstra, R. Lämmel, S. Staab, Deciding SHACL shape containment through description logics reasoning, in: J. Z. Pan, V. Tamma, C. d'Amato, K. Janowicz, B. Fu, A. Polleres, O. Seneviratne, L. Kagal (Eds.), The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I, volume 12506 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 366–383. URL: https://doi.org/10.1007/978-3-030-62419-4_21. doi:10.1007/978-3-030-62419-4_21.

[9] B. Bogaerts, M. Jakubowski, J. V. den Bussche, SHACL: A description logic in disguise, in: G. Gottlob, D. Inclezan, M. Maratea (Eds.), Logic Programming and Nonmonotonic Reasoning - 16th International Conference, LPNMR 2022, Genova, Italy, September 5-9, 2022, Proceedings, volume 13416 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 75–88. URL: https://doi.org/10.1007/978-3-031-15707-3_7. doi:10.1007/978-3-031-15707-3_7.

[10] B. Bogaerts, M. Jakubowski, J. V. den Bussche, Expressiveness of SHACL features and extensions for full equality and disjointness tests, Log. Methods Comput. Sci. 20 (2024). URL: https://doi.org/10.46298/lmcs-20(1:16)2024. doi:10.46298/LMCS-20(1:16)2024.

[11] A. Oudshoorn, M. Ortiz, M. Simkus, Reasoning with the core chase: the case of SHACL validation over ELHI knowledge bases, in: L. Giordano, J. C. Jung, A. Ozaki (Eds.), Proceedings of the 37th International Workshop on Description Logics (DL 2024), Bergen, Norway, June 18-21, 2024, volume 3739 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-3739/paper-7.pdf.

[12] W3C, Shape constraint language (SHACL), Technical Report. (2017). https://www.w3.org/TR/shacl.

[13] S. Rudolph, Foundations of description logics, in: A. Polleres, C. d'Amato, M. Arenas, S. Handschuh, P. Kroner, S. Ossowski, P. F. Patel-Schneider (Eds.), Reasoning Web. Semantic Technologies for the Web of Data - 7th International Summer School 2011, Galway, Ireland, August 23-27, 2011, Tutorial Lectures, volume 6848 of *Lecture Notes in Computer Science*, Springer, 2011, pp. 76–136. URL: https://doi.org/10.1007/978-3-642-23032-5_2. doi:10.1007/978-3-642-23032-5_2.

[14] C. Lutz, C. Areces, I. Horrocks, U. Sattler, Keys, nominals, and concrete domains, J. Artif. Intell. Res. 23 (2005) 667–726. URL: https://doi.org/10.1613/jair.1542. doi:10.1613/JAIR.1542.

[15] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, 2003.

[16] C. Lutz, U. Sattler, L. Tendera, The complexity of finite model reasoning in description logics, Inf. Comput. 199 (2005) 132–171. URL: https://doi.org/10.1016/j.ic.2004.11.002. doi:10.1016/J.IC.2004.11.002.

[17] S. Tobies, The complexity of reasoning with cardinality restrictions and nominals in expressive description logics, J. Artif. Intell. Res. 12 (2000) 199–217. URL: https://doi.org/10.1613/jair.705. doi:10.1613/JAIR.705.

[18] S. Tobies, Complexity results and practical algorithms for logics in knowledge representation, Ph.D. thesis, RWTH Aachen University, Germany, 2001. URL: http://sylvester.bth.rwth-aachen.de/dissertationen/2001/082/01_082.pdf.

[19] I. Pratt-Hartmann, Complexity of the two-variable fragment with counting quantifiers, J. Log. Lang. Inf. 14 (2005) 369–395. URL: https://doi.org/10.1007/s10849-005-5791-1. doi:10.1007/S10849-005-5791-1.

[20] R. Berger, The undecidability of the domino problem, 66, American Mathematical Soc., 1966.

[21] F. Baader, U. Sattler, Expressive number restrictions in description logics, J. Log. Comput. 9 (1999) 319–350. URL: https://doi.org/10.1093/logcom/9.3.319. doi:10.1093/LOGCOM/9.3.319.

[22] I. Horrocks, U. Sattler, S. Tobies, Practical reasoning for very expressive description logics, Log. J. IGPL 8 (2000) 239–263. URL: https://doi.org/10.1093/jigpal/8.3.239. doi:10.1093/JIGPAL/8.3.239.

[23] F. Baader, U. Sattler, Number restrictions on complex roles in description logics: A preliminary report., in: KR, 1996, pp. 328–339.

[24] S. Göller, M. Lohrey, C. Lutz, PDL with intersection and converse: satisfiability and infinite-state model checking, J. Symb. Log. 74 (2009) 279–314. URL: https://doi.org/10.2178/jsl/1231082313. doi:10.2178/JSL/1231082313.

[25] M. Schmidt-Schauß, Subsumption in KL-ONE is undecidable, in: R. J. Brachman, H. J. Levesque, R. Reiter (Eds.), Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR'89). Toronto, Canada, May 15-18 1989, Morgan Kaufmann, 1989, pp. 421–431.

[26] D. Calvanese, T. Eiter, M. Ortiz, Regular path queries in expressive description logics with nominals, in: C. Boutilier (Ed.), IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009, 2009, pp. 714–720. URL: http://ijcai.org/Proceedings/09/Papers/124.pdf.

[27] B. Bednarczyk, E. Kieronski, Finite entailment of local queries in the Z family of description logics, in: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022, AAAI Press, 2022, pp. 5487–5494. URL: https://doi.org/10.1609/aaai.v36i5.20487. doi:10.1609/AAAI.V36I5.20487.