Static Analysis of Recursive SHACL

Anouk Oudshoorn , Magdalena Ortiz , Mantas Šimkus
Institute of Logic and Computation, TU Wien
firstname.lastname@tuwien.ac.at

Abstract

SHACL (Shapes Constraint Language) expresses constraints on RDF data by means of so-called *shapes*. Its central service is *validation*: verifying whether a data graph complies with a SHACL document. But so far, there are no *static analysis services* to compare documents. In this paper, we study the following problem: decide whether all graphs that validate one SHACL document also validate another. Unlike previous works that have considered the implication of shape expressions only, we consider documents comprising (recursive) shape definitions and targets. We show that implication (a.k.a. containment) is undecidable under the supported and the stable model semantics, even for the fragment that uses the description logic $\mathcal{ALCIO}$ for shape expressions. Under the well-founded semantics, in surprising contrast, it is decidable in single exponential time. Our key technical contribution is a translation of SHACL under the well-founded semantics into the full hybrid μ-calculus, revealing a novel link between well-founded models and a fixed point modal logic, and a worst-case optimal automata-based decision procedure.[1]

1 Introduction

Graph-structured data is gaining widespread adoption because it avoids the need for rigid, predefined database schemas, which are difficult to design in settings where data is highly complex, incomplete, or subject to frequent structural change. In such environments, flexible graph data models like Knowledge Graphs (KGs) and Property Graphs are being increasingly adopted. To address the challenges arising in this graph-centric landscape, the W3C introduced SHACL (Shape Constraint Language) as a standard for expressing structural and semantic constraints over RDF graphs (Knublauch and Kontokostas 2017). The current SHACL recommendation defines semantics only for *non-recursive* SHACL constraints. Cyclic definitions (i.e., *recursion*) are envisioned in the W3C recommendation, but no semantics is defined, and validator implementations are free to handle recursion in an *ad hoc* manner. This problem has spurred research into suitable semantics for SHACL documents with recursive constraints, including the *supported model semantics* (Corman, Reutter, and Savkovic 2018), the *stable model semantics* (Andresel et al. 2020), and the *well-founded semantics* (Okulmus and Šimkus 2024). As their names suggest, these SHACL semantics are related to prominent semantics in Logic Programming. See Ahmetaj et al. (2026) for a more extensive discussion.

SHACL provides the means to define *shape expressions* that specify conditions that nodes in a graph are expected to satisfy. These definitions are paired with a set of *targets* in a SHACL document. The central service is *validation*, which consists of verifying whether a graph complies with a SHACL document. The computational complexity of this task has been studied for recursive SHACL under various semantics, discovering that in many settings it is tractable, yet intractability does arise in some surprising settings, e.g., for *stratified* constraints under the supported-model semantics (Corman, Reutter, and Savkovic 2018). A natural next step in the study of SHACL is understanding the computational properties of various *static analysis services*, such as *satisfiability* and *implication (a.k.a. containment)*. The difference between validation and the main challenge here is that these tasks are *data-independent*. The first task is to decide whether a given SHACL document is consistent, i.e., whether there exists at least one data graph that validates it. The second task is to decide whether all graphs that validate one document must necessarily validate another document. Checking implication in both directions allows to decide if two SHACL documents are equivalent. Such services are a stepping stone towards automated transformations and optimisation of SHACL documents.

Several insights into the working of SHACL have been achieved by examining its tight connection to Description Logics (DLs) (Pareti, Konstantinidis, and Mogavero 2022; Bogaerts, Jakubowski, and Van den Bussche 2022; Ortiz 2023; Di Stefano and Šimkus 2024). Specifically, since unrestricted SHACL shape expressions can be seen as concepts in very expressive DLs, like ones that support *role-value maps*, undecidability of satisfiability and implication testing in full expressive SHACL can be inferred (Pareti, Konstantinidis, and Mogavero 2022). Identifying restricted settings in which some static analysis problems become decidable and to infer upper bounds on their complexity can be done by drawing from the DL literature. For instance, Leinberger et al. (2020) provide some positive results on reasoning about containment of shape expressions instead of fully-fledged SHACL documents. Furthermore, Pareti,

[1] This is the long version of this work accepted at KR2026.

Konstantinidis, and Mogavero (2022) provide some positive results for satisfiability of recursive SHACL constraints under the supported-model semantics, while Oudshoorn (2025) extends these results further and shows that, for the supported model semantics, full document satisfiability is reducible to satisfiability of a single constraint. However, our understanding of satisfiability and implication in recursive SHACL is still very limited. In fact, to the best of our knowledge, no attempts have been made to encode this problem into decidable logic formalisms, or solve it by other means.

Thus, this paper provides the first dedicated study of the computational complexity of implication of SHACL documents in the presence of recursive constraints. That is, we study the problem of deciding, given two SHACL documents $\mathcal{S}_1$ and $\mathcal{S}_2$ containing shape definitions and targets for validation, whether every graph that validates $\mathcal{S}_1$ also validates $\mathcal{S}_2$. Our first major result is that, if one considers the supported or the stable model semantics, this problem is undecidable even when the language for shape expressions is restricted to the SHACL fragment that corresponds to the DL $\mathcal{ALCIO}$. This is somewhat surprising, but becomes even more intriguing in the light of our second main result: under the well-founded semantics, this problem is not only decidable, but is feasible in single exponential time and thus not harder than deciding satisfiability of a single shape expression.

The main stepping stone to obtaining our positive results is a translation of SHACL documents under the well-founded semantics into formulas of the full hybrid μ-calculus. This translation establishes a novel connection between logic with least and greatest fixed points on the one hand, and the well-founded semantics on the other. Interestingly, capturing the well-founded semantics requires limited alternation of fixed point operators. This connection is, to our knowledge, novel and of independent interest. We note that some initial works on the well-founded semantics for logic programs have already established a connection for logics with fixed points (Van Gelder, Ross, and Schlipf 1991; Van Gelder 1993). Our setting here is different: their monotonic operator is composed of two anti-monotonic fixed point operators. That is, the first operator produces an overestimate of the set of negative conclusions, whereas the second is an underestimate. This alternating, estimating behaviour is absent in our work, in the sense that we translate (directly) into (fragments of) the full hybrid μ-calculus. We rely on the insights of our more constructive translation to derive the tight complexity bounds.

That is, our translation of a SHACL document into a μ-calculus formula unfolds the rules defining shapes into a single formula, potentially leading to an exponential blow-up. We therefore also provide an automata-based algorithm for evaluating (the μ-calculus translation of) a SHACL document that runs in time bounded by a single exponential function of the size of the input document. This allows us to establish optimal complexity bounds for SHACL static validation services in a range of fragments.

2 Preliminaries

Data Graphs. Let N_C, N_R and N_I denote countably infinite, mutually disjoint sets of *concept names*, *role names*, and *individuals* respectively, such that $\top \in N_C$. 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 $A(a)$ or $p(a, a')$, for $A \in N_C$, $p \in N_R$ and $\{a, a'\} \subseteq N_I$. A *data graph* $\mathcal{G}$ is a finite set of atoms. Let $r^{\mathcal{G}} := \{(a, a') \in N_I \mid r(a, a') \in \mathcal{G}\}$ and $A^{\mathcal{G}} := \{a \in N_I \mid A(a) \in \mathcal{G}\}$. Let $\Delta^{\mathcal{G}}$ denote the set of all individuals that appear in a data graph $\mathcal{G}$.

Shape Constraint Language (SHACL). We introduce SHACL following the line first set out by Corman, Reutter, and Savkovic (2018). We start by defining *shape expressions*. We do not define them for full SHACL, but only the fragments considered in this paper. To conveniently talk about these fragments, we borrow notation from Description Logics. Let N_S denote a countably infinite set of *shape names* that is disjoint from $N_C \cup N_R \cup N_I$. Specifically, shape expressions φ are build using the following grammar:

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

where $s \in N_S$, $a \in N_I$, $A \in N_C$, and $r \in \overline{N}_R$. We refer to this fragment as $\mathcal{ALCIO}$ SHACL, using the DL naming convention. If in the above fragment, r is restricted to N_R, that is, we can only access roles in one direction, we call this $\mathcal{ALCO}$ SHACL. If, on the other hand, the fragment does not use any $a \in N_I$, we are in $\mathcal{ALCI}$ SHACL. If both restrictions apply, this is referred to as $\mathcal{ALC}$ SHACL.

A *shape constraint* is an expression of the form $s \leftarrow \varphi$, for $s \in N_S$ and φ a shape expression. We call s the *head* of the constraint. Given a set of shape constraints $\mathcal{C}$, 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. A *shape atom* has the form $s(a)$ with $s \in N_S$ and $a \in N_I$, and we call $\neg s(a)$ *negated shape atom*; a *shape literal* is a possibly negated shape atom.

Let

$$\begin{aligned} sub(\psi) &:= \{\psi\} \\ sub(\neg s) &:= \{\neg s, s\} \\ sub(\varphi * \varphi') &:= \{\varphi * \varphi'\} \cup sub(\varphi) \cup sub(\varphi') \\ sub(\circ r.\varphi) &:= \{\circ r.\varphi\} \cup sub(\varphi) \end{aligned}$$

for $\psi \in N_C \cup N_I \cup N_S$, $* \in \{\wedge, \vee\}$ and $\circ \in \{\exists, \forall\}$. For any constraint set $\mathcal{C}$, let $sub(\mathcal{C}) := \{sub(\varphi) \mid s \leftarrow \varphi \in \mathcal{C}\}$.

A *target* is a pair (ℓ, s), where $\ell \in N_I \cup N_C \cup \overline{N}_R$. A SHACL document is a pair $(\mathcal{C}, \mathcal{T})$, where $\mathcal{C}$ is a set of shape constraints and $\mathcal{T}$ is a set of targets. With a slight abuse of notation, we sometimes use a shape atom $s(\ell)$ to specify a target (ℓ, s), and write $(\mathcal{C}, s(\ell))$ instead of $(\mathcal{C}, \{(\ell, s)\})$.

The semantics of SHACL is based on the notion of a *shape assignment*, defined as a set S of shape shape literals that are not contradictory, that is, there are no $\{s(a), \neg s(a)\} \subseteq S$. We view such shape assignments as three-valued: intuitively, $s(a) \in S$ means that s is

validated at a, $\neg s(a) \in S$ means that s is not validated, and $\{s(a), \neg s(a)\} \cap S = \emptyset$ means that the validation of s at a is *undefined* in S. We say S is *total*, if for every shape name s and individual a that appear in S, we have $s(a) \in S$ or $\neg s(a) \in S$.

We say $\mathcal{G}$ validates a set of targets $\mathcal{T}$ under a shape assignment S if the following are satisfied:

1. $s(a) \in S$, for every $(a, s) \in \mathcal{T}$,
2. $s(a) \in S$, for every $(A, s) \in \mathcal{T}$ and $a \in A^{\mathcal{G}}$, and
3. $s(a) \in S$, for every $(r, s) \in \mathcal{T}$ and $(a, a') \in r^{\mathcal{G}}$.

To define validation for a SHACL document $(\mathcal{C}, \mathcal{T})$, we still need to give semantics to shape expressions under a given shape assignment. For recursive SHACL, three ways to do this have been advocated in the literature: the well-founded (Okulmus and Šimkus 2024), supported (Corman, Reutter, and Savkovic 2018), and stable semantics (Andresel et al. 2020). The well-founded semantics yields a unique assignment for each $\mathcal{G}$ and $\mathcal{C}$. In contrast, the supported and the stable semantics both yield a set that may contain several *total* assignments (or none at all); in such cases, one adopts *brave* validation.

Definition 1. *A* semantics $\mathbb{S}$ *for SHACL is a function that maps each pair of a graph and a set of constraints to a set of assigments. We say that $\mathcal{G}$ satisfies $(\mathcal{C}, \mathcal{T})$ under $\mathbb{S}$ if $\mathcal{G}$ validates $\mathcal{T}$ under some assignment in $\mathbb{S}(\mathcal{G}, \mathcal{C})$. We say that a node a in $\mathcal{G}$ satisfies a shape s w.r.t. $\mathcal{C}$ under $\mathbb{S}$ if $\mathcal{G}$ satisfies $(\mathcal{C}, s(a))$. If some node of $\mathcal{G}$ satisfies s w.r.t. $\mathcal{C}$, we say that $\mathcal{G}$* satisfies s w.r.t. $\mathcal{C}$*, or for simplicity, just $\mathcal{G}$* satisfies $(\mathcal{C}, s)$.

We define next the well-founded semantics, which is the focus of this paper, and briefly recall the supported semantics. For the full definition of the stable semantics, we refer the reader to (Andresel et al. 2020).

Recall that, based on the allowed syntax for SHACL documents, concrete individuals may explicitly appear inside shape expression or targets. To make presentation simpler, when we talk about checking if a graph $\mathcal{G}$ satisfies a document $(\mathcal{C}, \mathcal{T})$, we assume that $\mathcal{G}$ is *compatible* with $(\mathcal{C}, \mathcal{T})$. By this we mean that all individuals that appear in $(\mathcal{C}, \mathcal{T})$ also appear in $\mathcal{G}$. Validation for incompatible $\mathcal{G}$ and $(\mathcal{C}, \mathcal{T})$ is considered undefined. When we quantify over possible graphs $\mathcal{G}$ for validation or non-validation w.r.t. a document $(\mathcal{C}, \mathcal{T})$, the quantification is always limited to graphs that are compatible with $(\mathcal{C}, \mathcal{T})$.

Well-founded Semantics. For a given data graph $\mathcal{G}$ and shape assignment S, we define two functions $\lfloor\cdot\rfloor_S^{\mathcal{G}}$ and $\lceil\cdot\rceil_S^{\mathcal{G}}$ that map shape expressions to sets of nodes as described in Figure 1. Let $\mathcal{C}(s) := \varphi$, where $s \leftarrow \varphi \in \mathcal{C}$ is the unique constraint in $\mathcal{C}$ with head s. Intuitively, assuming that the literals in S are true, $\lfloor\varphi\rfloor_S^{\mathcal{G}}$ is the set of nodes where φ is certainly validated, whereas $\lceil\varphi\rceil_S^{\mathcal{G}}$ is the set of nodes for which validation is possible, as it is not prevented by S. We can use $\lfloor\cdot\rfloor_S^{\mathcal{G}}$ to infer positive shape literals in the well-founded model: if $a \in \lfloor\varphi\rfloor_S^{\mathcal{G}}$, we can infer $s(a)$. In contrast, from $\lceil\cdot\rceil_S^{\mathcal{G}}$ we can infer negated shape literals: if $a \notin \lceil\varphi\rceil_S^{\mathcal{G}}$, then we can infer $\neg s(a)$. These inferred literals are added to the well-founded model incrementally, using the following one-step operators. For the positive inferences, the operator is

$$T_{\mathcal{G},\mathcal{C}}(S) := \{s(a) \mid s \leftarrow \varphi \in \mathcal{C}, a \in \lfloor\varphi\rfloor_S^{\mathcal{G}}\}.$$

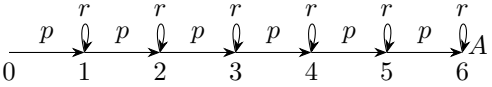
For inferring negative atoms, we rely on *unfounded sets*. For a set U of shape atoms, we let $\neg.U := \{\neg s(a) \mid s(a) \in U\}$. Given a shape assignment S, a data graph $\mathcal{G}$ and a set of constraints $\mathcal{C}$, we call a set of shape atoms U *unfounded* w.r.t. S, $\mathcal{G}$ and $\mathcal{C}$ if $a \notin \lceil\mathcal{C}(s)\rceil_{S \cup \neg.U}^{\mathcal{G}}$ for all $s(a) \in U$. As being unfounded is preserved under union, there exists a unique *greatest unfounded set* w.r.t. S, $\mathcal{G}$ and $\mathcal{C}$ that is not subsumed by a strictly larger unfounded set w.r.t. S, $\mathcal{G}$ and $\mathcal{C}$. Let $U_{\mathcal{G},\mathcal{C}}$ be a function that maps each shape assignment S to the greatest unfounded set w.r.t. S, $\mathcal{G}$ and $\mathcal{C}$.

Both positive and negative consequences are combined in the operator $W_{\mathcal{G},\mathcal{C}}$. Like $T_{\mathcal{G},\mathcal{C}}$ and $U_{\mathcal{G},\mathcal{C}}$, it is a function that maps shape assignments to shape assignments.

$$W_{\mathcal{G},\mathcal{C}}(S) := T_{\mathcal{G},\mathcal{C}}(S) \cup \neg.U_{\mathcal{G},\mathcal{C}}(S). \tag{1}$$

We say a function f is *monotone* if $x \subseteq y$ implies $f(x) \subseteq f(y)$. The above *immediate consequence operator* $W_{\mathcal{G},\mathcal{C}}$ is a monotone function, hence it has a unique least fixed point. We denote this fixed point $WF_{\mathcal{G},\mathcal{C}}$ and call it the *well-founded model* of $\mathcal{G}$ and $\mathcal{C}$.

Example 1. *Consider the constraint set $\mathcal{C} = \{s \leftarrow A \vee \exists p.\neg t, t \leftarrow \neg s \vee \exists r.t\}$. The SHACL document $(\mathcal{C}, \{s(0)\})$ is satisfiable in, for instance, the following structure $\mathcal{G}_1$.*



To see this, observe the following step-by-step computation of the least fixed point $WF_{\mathcal{G}_1,\mathcal{C}}$.

$$W_{\mathcal{G}_1,\mathcal{C}}(\emptyset) = \{s(6)\} \cup \emptyset$$
$$W^2_{\mathcal{G}_1,\mathcal{C}}(\emptyset) = \{s(6)\} \cup \{\neg t(6)\}$$
$$W^3_{\mathcal{G}_1,\mathcal{C}}(\emptyset) = \{s(6), s(5)\} \cup \{\neg t(6)\}$$
$$W^4_{\mathcal{G}_1,\mathcal{C}}(\emptyset) = \{s(6), s(5)\} \cup \{\neg t(6), \neg t(5)\}$$
$$\cdots$$

Note that if all p-paths from 0 to an A are infinitely long (and there exists one), this least fixed point will not terminate in a finite number of steps.

Supported Model Semantics. A supported model for constraint set $\mathcal{C}$ and a graph $\mathcal{G}$ is a total shape assignments S that satisfies $s^{\mathcal{G},S} = \lfloor\varphi\rfloor_S^{\mathcal{G}}$ for every $s \leftarrow \varphi \in \mathcal{C}$. We say $\mathcal{G}$ validates a SHACL document $(\mathcal{C}, \mathcal{T})$ under the *Supported Model Semantics*, if there is some supported model S of $\mathcal{C}$ and a graph $\mathcal{G}$ that satisfies $\mathcal{T}$. Note that above we used $\lfloor\cdot\rfloor_S^{\mathcal{G}}$. However, both $\lfloor\cdot\rfloor_S^{\mathcal{G}}$ and $\lceil\cdot\rceil_S^{\mathcal{G}}$ can be used interchangeably here, since $\lfloor\varphi\rfloor_S^{\mathcal{G}} = \lceil\varphi\rceil_S^{\mathcal{G}}$, for any φ, $\mathcal{G}$, and total S.

Example 2. *Consider the same constraint set as in Example 1, but now for the following structure $\mathcal{G}_2$.*

$$\lfloor s \rfloor^{\mathcal{G}}_S := \{a \mid s(a) \in S\} \qquad \lceil s \rceil^{\mathcal{G}}_S := \{a \in \Delta^{\mathcal{G}} \mid \neg s(a) \notin S\}$$

$$\lfloor \neg s \rfloor^{\mathcal{G}}_S := \{a \mid \neg s(a) \in S\} \qquad \lceil \neg s \rceil^{\mathcal{G}}_S := \{a \in \Delta^{\mathcal{G}} \mid s(a) \notin S\}$$

$$[a]^{\mathcal{G}}_S := \{a^{\mathcal{G}}\} \qquad [A]^{\mathcal{G}}_S := A^{\mathcal{G}} \qquad [\cdot] \in \{\lfloor \cdot \rfloor, \lceil \cdot \rceil\}$$

$$[\varphi \vee \varphi']^{\mathcal{G}}_S := [\varphi]^{\mathcal{G}}_S \cup [\varphi']^{\mathcal{G}}_S \qquad [\varphi \wedge \varphi']^{\mathcal{G}}_S := [\varphi]^{\mathcal{G}}_S \cap [\varphi']^{\mathcal{G}}_S \qquad [\cdot] \in \{\lfloor \cdot \rfloor, \lceil \cdot \rceil\}$$

$$[\forall r.s]^{\mathcal{G}}_S := \left\{a \in \Delta^{\mathcal{G}} \mid \forall a' \in \Delta^{\mathcal{G}} : (a, a') \in r^{\mathcal{G}} \text{ implies } a' \in [s]^{\mathcal{G}}_S\right\} \qquad [\cdot] \in \{\lfloor \cdot \rfloor, \lceil \cdot \rceil\}$$

$$[\exists r.s]^{\mathcal{G}}_S := \left\{a \in \Delta^{\mathcal{G}} \mid \text{ exists } a' \text{ s.t. } (a, a') \in r^{\mathcal{G}} \text{ and } a' \in [s]^{\mathcal{G}}_S\right\} \qquad [\cdot] \in \{\lfloor \cdot \rfloor, \lceil \cdot \rceil\}$$

Figure 1: Evaluating shape expressions: upper and lower bounds.

As $WF_{\mathcal{G}_2,\mathcal{C}}(\emptyset) = \emptyset$, we find that $\mathcal{G}_2$ and $\mathcal{C}$ only have empty well-founded models, whereas there are two supported models S_1 and S_2:

$$S_1 = \{s(0), s(1), \neg t(0), \neg t(1)\}$$
$$S_2 = \{\neg s(0), \neg s(1), t(0), t(1)\}.$$

Stratified Constraints. Derived from the well-known class of stratified programs (Apt, Blair, and Walker 1988), stratified SHACL can be defined in the following way.

Definition 2. *We say a shape name s is* defined *in a set $\mathcal{C}$ of constraints if $s \leftarrow \varphi \in \mathcal{C}$ for some φ. A set $\mathcal{C}$ of constraints is* stratified *if it can be partitioned into sets $\mathcal{C}_0, \ldots, \mathcal{C}_k$ such that, for all $0 \leq i \leq k$, the following hold:*

1. *If $i < k$ and $s' \in sub(\varphi)$ for some $s \leftarrow \varphi \in \mathcal{C}_i$, then s' is defined in $\mathcal{C}_0 \cup \ldots \cup \mathcal{C}_i$.*
2. *If $\neg s' \in sub(\varphi)$ for some $s \leftarrow \varphi \in \mathcal{C}_i$, then s' is defined in $\mathcal{C}_0 \cup \ldots \cup \mathcal{C}_{i-1}$.*

A set of constraints is stratified *if it admits a stratification.*

For stratified constraints, $WF_{\mathcal{G},\mathcal{C}}$ is total, it is a supported model, and it is the unique stable model of $\mathcal{G}$ and $\mathcal{C}$; it is sometimes called the *least* or *perfect* model of $\mathcal{G}$ and $\mathcal{C}$.

Normal Form. To simplify presentation, without loss of generality, we will mostly focus on SHACL constraints in which there is no nesting of constructors. Specifically, we focus on constraints that have one of the following forms:

$$s \leftarrow A \qquad s \leftarrow a \qquad s \leftarrow \neg s' \qquad s \leftarrow \exists r.s'$$
$$s \leftarrow \forall r.s' \qquad s \leftarrow s' \wedge s'' \qquad s \leftarrow s' \vee s''$$

where $A \in N_C$, $a \in N_I$, $r \in \overline{N}_R$, and $\{s, s', s''\} \subseteq N_S$. For all semantics we consider, transforming an arbitrary SHACL document into one where all constraints are in normal form while preserving validation is straightforward. Intuitively, this is because whenever a complex subformula ψ occurs in a shape constraint $s \leftarrow \varphi$, we can simply replace ψ in φ by a fresh shape name s' and add $s' \leftarrow \psi$ to the constraint collection. See, e.g. the constructions by Andresel et al. (2020) and Oudshoorn, Ortiz, and Šimkus (2024).

3 Static Analysis of SHACL Specifications

In this paper we study the following decision problems for the different semantics $\mathbb{S}$ described above.

(Finite) document satisfiability: Given a SHACL document $(\mathcal{C}, \mathcal{T})$, decide whether there exists a (finite) graph that satisfies $(\mathcal{C}, \mathcal{T})$ under $\mathbb{S}$.

Finite model property: Is it the case that, under $\mathbb{S}$, every satisfiable SHACL document is finitely satisfiable?

(Finite) document implication: Given two SHACL documents $(\mathcal{C}_1, \mathcal{T}_1)$ and $(\mathcal{C}_2, \mathcal{T}_2)$, decide whether every (finite) graph that satisfies $(\mathcal{C}_1, \mathcal{T}_1)$ under $\mathbb{S}$ also satisfies $(\mathcal{C}_2, \mathcal{T}_2)$.

We also consider the special cases where we target validation of one specific shape name:

(Finite) shape satisfiability w.r.t. constraints: Given a set of constraints $\mathcal{C}$ and a shape name s, decide whether there exists a (finite) graph that satisfies $(\mathcal{C}, s)$ under $\mathbb{S}$.

(Finite) shape implication w.r.t. constraints: Given shape constraints $\mathcal{C}_1$ and $\mathcal{C}_2$ and shape names s_1 and s_2, decide whether every node that satisfies $(\mathcal{C}_1, s_1)$ under $\mathbb{S}$ in a graph $\mathcal{G}$ also satisfies $(\mathcal{C}_2, s_2)$.

To our knowledge, document and shape satisfiability are the only problems that have been studied before, except for one work that studied implication (sometimes called containment) for SHACL, but in a more restricted setting; Leinberger et al. (2020) studied satisfiability and implication problem for shape expressions alone, with no constraints. Their problem corresponds to deciding whether $(\{s \leftarrow \varphi_1\}, s)$ is satisfiable, and whether $(\{s_1 \leftarrow \varphi_1\}, s_1)$ implies $(\{s_2 \leftarrow \varphi_1\}, s_2)$ for given φ_1, φ_2.

For the supported model semantics, document satisfiability has been considered by Pareti, Konstantinidis, and Mogavero (2022) and Oudshoorn (2025). In this semantics, these problems naturally reduce to satisfiability and logical implication in fragments of classical first-order logic (FO), or equivalently, to concept satisfiability and subsumption w.r.t. TBoxes in Description Logics (DLs). Pareti, Konstantinidis, and Mogavero obtained many undecidability and some decidability results by analysing the target FO fragments, which were then significantly expanded and tightened in terms of complexity by Oudshoorn building on DLs. Some of the results mentioned

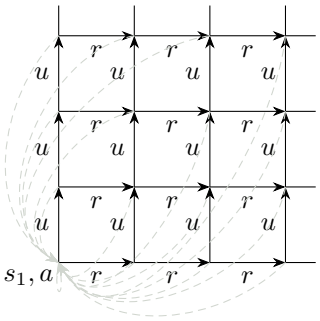


Figure 2: Infinite grid that, after adding s as label to every node, shows undecidability of shape implication w.r.t. constraints under the supported model semantics. The grey dashed arrows indicate the extension of the role l.

above consider both finite and unrestricted models; others focus on finite ones. In some cases, the finite model property is inferred by establishing a connection to logics that enjoy it. We refer to Oudshoorn (2025) for a more extensive discussion.

We are not aware of any works considering satisfiability and containment under the stable or well-founded semantics, but an EXPTIME upper bound for shape satisfiability w.r.t. constraints under the stable semantics for the $\mathcal{ALCI}$ fragment can be inferred from the work of Di Stefano and Šimkus (2024) on DL terminologies under the stable (a.k.a. *equilibrium*) semantics.

Theorem 1. *Shape implication w.r.t. constraints for (stratified) $\mathcal{ALCIO}$ SHACL is undecidable, under the supported model semantics.*

Proof. We prove this by reducing the tiling problem, a well-known undecidable problem (Berger 1966), to our setting. To achieve this reduction we construct two shape pairs such that $(\mathcal{C}, s_1)$ implies $(\mathcal{C}', s')$ iff there exists a grid that can be decorated with a tiling $t : \mathbb{N} \times \mathbb{N} \to D$.

That is, let $\mathcal{C} = \{s_1 \leftarrow a \wedge s, s \leftarrow \exists l.a \wedge \exists u.s \wedge \exists r.s \wedge \bigvee_{d \in D} s_d \wedge \neg s_\bot\} \cup \mathcal{C}_{tiles}$, where $\mathcal{C}_{tiles}$ contains the constraints ensuring the correct horizontal and vertical relations between the tiles, and sends the error shape $s_\bot$ back in case more than one tile was assigned to one node - as this is straightforward to construct, we will focus on the grid construction in the rest of this proof. Furthermore, we let $\mathcal{C}' = \{s_a \leftarrow s_a, s_b \leftarrow s_b, s' \leftarrow \exists l^-.(\exists r \exists u.s_a \wedge \exists u.\exists r.s_b) \wedge \forall l^-.((s_a \vee s_b) \wedge \neg(s_a \wedge s_b)\}$.

Deciding whether $(\mathcal{C}, s_a)$ implies $(\mathcal{C}', s')$ is equivalent to deciding whether there exists a data graph $\mathcal{G}$ such that $\mathcal{G}$ satisfies $(\mathcal{C}, s_a)$, and not $(\mathcal{C}', s')$. Note that to satisfy $(\mathcal{C}, s_a)$ it must be possible to map an infinite binary tree (with u and r edges) into $\mathcal{G}$, with every node linking back via l to the root a. For $\mathcal{G}$ to not satisfy $(\mathcal{C}', s')$ means it is not possible to properly decorate $\mathcal{G}$ with shape names. If we take a closer look at $\mathcal{C}'$, it is immediate that $s_a \leftarrow s_a$ and $s_b \leftarrow s_b$ cannot cause any problem. Therefore, the impossibility must lie in $s' \leftarrow \exists l^-.(\exists r \exists u.s_a \wedge \exists u \exists r.s_b) \wedge \forall l^-.(s_a \vee s_b \wedge \neg(s_a \wedge s_b))$. For a node to not be an s' means that at least $\exists l^-.(\exists r \exists u.s_a \wedge \exists u \exists r.s_b)$ or $\forall l^-.((s_a \vee s_b) \wedge \neg(s_a \wedge s_b)$ cannot be realised on the given structure. As the latter, an exclusive-or labelling nodes with either an s_a or an s_b, cannot be blocked in any structure, it is $\exists l^-.(\exists r \exists u.s_a \wedge \exists u.\exists r.s_b)$ that must be blocked. As every node must have a u- and r-successor, and as s_a and s_b are mutually exclusive, this can only be blocked if for every node reachable by l^-, we find there can be only one node reachable by both ru or ur. In other words, $(\mathcal{C}, s_a)$ does not imply $(\mathcal{C}', s')$ iff there exists a grid that can be decorated with tiles. □

Note that the problem remains undecidable even if we require the set of constraints to be the same in both documents: in the proof, we can replace both $\mathcal{C}$ and $\mathcal{C}'$ by their union. Not surprisingly, the undecidability applies also to the stable semantics, but only in the non-stratified setting.

Theorem 2. *Shape implication w.r.t. constraints for $\mathcal{ALCIO}$ SHACL is undecidable, under the stable model semantics.*

To see this, update $\mathcal{C}'$ to $\mathcal{C}' = \{s_a \leftarrow \neg s_b, s_b \leftarrow \neg s_a, s' \leftarrow \exists l^-.(\exists r \exists u.s_a \wedge \exists u \exists r.s_b)\}$. Here, we use the non-stratifiedness of $\mathcal{C}'$ to enforce the choice of an s_a or s_b label. The rest of the proof works in the same way.

Thus, we shift our attention to SHACL containment under the well-founded semantics in the rest of this article, and find we can recover decidability of containment. We note it will sometimes be convenient to treat implication as a special case of containment.

Proposition 1. *A document $(\mathcal{C}, \mathcal{T})$ is (finitely) satisfiable under $\mathbb{S}$ iff $(\mathcal{C}, \mathcal{T})$ does not (finitely) imply $(\emptyset, \emptyset)$ under $\mathbb{S}$.*

4 Full Hybrid μ-calculus

We first provide some preliminary definitions for the full hybrid μ-calculus.[2] Let N_V be a set of *variables*, disjoint from N_I, N_C and N_R. Then, a full hybrid μ-calculus formula is given by

$$\Phi ::= A \mid \neg A \mid a \mid \neg a \mid X \mid \Phi \wedge \Phi \mid \Phi \vee \Phi \mid [r]\Phi \mid \langle r \rangle \Phi \mid \mu X.\Phi \mid \nu X.\Phi,$$

for $A \in N_C, a \in N_I, X \in N_V, r \in \overline{N}_R$. Let $\top := A \vee \neg A$, and $\bot := A \wedge \neg A$ for a $A \in N_C$.

The set of *subformulas* is defined recursively by setting $sub(C) := \{C\}$ and $sub(\neg C) := \{\neg C\}$ for each $C \in N_I \cup N_C \cup N_V$, $sub(\Phi * \Phi') := \{\Phi * \Phi'\} \cup sub(\Phi) \cup sub(\Phi')$, for each $* \in \{\wedge, \vee\}$, $sub(@\Phi) := \{@\Phi\} \cup sub(\Phi)$ for each $@ \in \{\langle r \rangle, [r] \mid r \in \overline{N}_R\}$, and $sub(\sigma X.\Phi) := \{\sigma X.\Phi\} \cup sub(\Phi)$, for $\sigma \in \{\mu, \nu\}$. We say Ψ appears or is contained in Φ is $\Psi \in sub(\Phi)$.

A variable $X \in N_V$ is a *free variable* if X appears outside of the scope of a fixed point operator σX, for $\sigma \in \{\mu, \nu\}$. A formula is called *closed* if it contains no free variables.

Given an data graph $\mathcal{G}$, the semantics $||\Phi||^{\mathcal{G}}_V \subseteq \Delta^{\mathcal{G}}$ of a closed formula Φ is defined as in Figure 3. Here, $V : N_V \to$

[2] μ-calculus with inverse programs and nominals; we follow the naming convention of (Bonatti et al. 2008).

$2^{\Delta^{\mathcal{G}}}$ is a function that maps each fixed point variable X to a set of nodes. During evaluation, the function V is updated as follows: $V[X \to U](X) := U$, and $V[X \to U](Y) := V(Y)$ for $Y \neq X$.

For a closed formula Φ, we may use $\mathcal{G}, c \models \Phi$ as a shorthand for $c \in ||\Phi||^{\mathcal{G}}_V$. Note that because each variable X always occurs in the scope of a fixed point operator σX in a closed formula, it is unnecessary to provide a valuation function V.

Although unusual, we allow fixed point operators $\sigma X.\Psi$ such that X does not occur in Ψ. We call a formula *clean* if there do not appear such fixed point operators in the formula. Note that if X is not contained in Ψ, removing or adding σX does not influence the semantics of the formula.

Proposition 2. *For each fully hybrid μ-calculus formula, and each data graph $\mathcal{G}$ and node $c \in N_I$, we find*

$$\mathcal{G}, c \models \Phi \quad \textit{iff} \quad \mathcal{G}, c \models cln(\Phi).$$

A well-known classical result is that fixed points can be calculated by iteratively updating the valuation for X with the result of the previous approximation (see, e.g. Arnold and Niwinski (2001)). That is, the (non-transfinite version of the) approximations of the least and greatest fixed point are given by

$$\begin{aligned} ||\mu^0 X.\Phi||^{\mathcal{G}}_V &:= \emptyset, \\ ||\mu^{\alpha+1} X.\Phi||^{\mathcal{G}}_V &:= ||\Phi||^{\mathcal{G}}_{V[X \mapsto ||\mu^\alpha X.\Phi||^{\mathcal{G}}_V]} \\ ||\mu^{\omega} X.\Phi||^{\mathcal{G}}_V &:= \bigcup_{\alpha<\omega} ||\mu^\alpha X.\Phi||^{\mathcal{G}}_V, \end{aligned}$$

and

$$\begin{aligned} ||\nu^0 X.\Phi||^{\mathcal{G}}_V &:= \Delta^{\mathcal{G}}, \\ ||\nu^{\alpha+1} X.\Phi||^{\mathcal{G}}_V &:= ||\Phi||^{\mathcal{G}}_{V[X \mapsto ||\nu^\alpha X.\Phi||^{\mathcal{G}}_V]}, \\ ||\nu^{\omega} X.\Phi||^{\mathcal{G}}_V &:= \bigcap_{\alpha<\omega} ||\nu^\alpha X.\Phi||^{\mathcal{G}}_V. \end{aligned}$$

These approximations lead to the following infinite increasing resp. decreasing chains

$$\begin{aligned} &||\mu^0 X.\Phi||^{\mathcal{G}}_V \subseteq ||\mu^1 X.\Phi||^{\mathcal{G}}_V \subseteq \ldots \subseteq ||\mu^\alpha X.\Phi||^{\mathcal{G}}_V \subseteq \ldots \\ &||\nu^0 X.\Phi||^{\mathcal{G}}_V \supseteq ||\nu^1 X.\Phi||^{\mathcal{G}}_V \supseteq \ldots \supseteq ||\nu^\alpha X.\Phi||^{\mathcal{G}}_V \supseteq \ldots \end{aligned}$$

Moreover, as we assume $|\Delta^{\mathcal{G}}| \leq \omega$, we can conclude the following.

Proposition 3. *For all full hybrid μ-calculus formulas $\sigma X.\Phi$, we find*

$$||\sigma^\omega X.\Phi||^{\mathcal{G}}_V = ||\sigma X.\Phi||^{\mathcal{G}}_V.$$

5 Translating SHACL into μ-calculus

The goal of this section is to provide a translation that takes as input a SHACL specification $\mathcal{C}$ and target shape name s and produces a μ-calculus formula $\Phi_{\mathcal{C},s}$ such that, for every graph $\mathcal{G}$, the nodes that validate s with respect to $\mathcal{C}$ are exactly the nodes that satisfy $\Phi_{\mathcal{C},s}$.

The translation is presented in Figure 4 and uses two mutually recursive functions, $tr^+_{S,\mathcal{C}}$ and $tr^-_{S,\mathcal{C}}$. The desired $\Phi_{\mathcal{C},s}$ will be the (cleaned version of) $tr^+_{\emptyset,\mathcal{C}}(s)$. Recall that the well-founded model is defined as the least fixed point of the operator $W_{\mathcal{G},\mathcal{C}}$ in (1); it is thus not surprising that $tr^+_{\emptyset,\mathcal{C}}(s)$ takes the form of a least fixed point $\mu X\varphi$. The positive part $tr^+_{S,\mathcal{C}}$ will mimic the derivation of the positive atoms that are introduced by $T_{\mathcal{G},\mathcal{C}}(S)$ during the well-founded model computation, while the $tr^-_{S,\mathcal{C}}$ part will be the counterpart of the negative atoms added by $\neg.U_{\mathcal{G},\mathcal{C}}(S)$. It is this second part that is more interesting. Recall that $U_{\mathcal{G},\mathcal{C}}(S)$ is the *greatest* unfounded set, that is, the largest set of atoms that do not break a certain condition ($a \notin \lceil \mathcal{C}(s) \rceil^{\mathcal{G}}_{S \cup \neg.U}$ for all $s(a) \in U$). This seems to naturally correspond to a greatest fixed point, but we need a bit more than just a ν: note that when deciding whether $b \in \lceil s \rceil^{\mathcal{G}}_{S \cup \neg.U}$, we do not just check whether $\neg s(b) \notin \neg.U$, but also whether $\neg s(b) \notin S$, that is, the extension of the set is affected not only by the set U that is being computed as a greatest fixed point, but also by the set S that are computed by the least fixed point corresponding to $W_{\mathcal{G},\mathcal{C}}(S)$. Therefore, our translation produces a formula with limited alternation: it takes the form $\mu X.\varphi(X)$, where $\varphi(X)$ may contain subformulas of the form $\nu Y.\psi(X, Y)$. There are, however, no fixed points nested within greatest fixed points: observe that the subscript S is changed to $pos(S)$ when we move to the negative part $tr^+_{S,\mathcal{C}}$ of the translation.

In the following example, we illustrate that this limited alternation is needed. That is, the alternation-free μ-calculus is not expressive enough to capture the well-founded semantics.

Example 3. *Recall the constraints $\mathcal{C} = \{s \leftarrow A \vee \exists p.\neg t, t \leftarrow \neg s \vee \exists r.t\}$ discussed in Example 1. Then*

$$tr^+_{\emptyset,\mathcal{C}}(s) = \mu X_s.(A \vee \langle p \rangle \nu X_{\bar{t}}.(X_s \wedge [r] X_{\bar{t}})),$$

which is a formula expressing a property not expressible in alternation free μ-calculus. Note that if we evaluate $tr^+_{\emptyset,\mathcal{C}}(s)$ over $\mathcal{G}_1$, we find $||tr^+_{\emptyset,\mathcal{C}}(s)||^{\mathcal{G}_1}_V = \{0, \ldots, 6\}$.

We can now prove the correctness of our translation.

Theorem 3. *For each shapes pair $(\mathcal{C}, s)$, each data graph $\mathcal{G}$, and each node c of $\mathcal{G}$, we have*

$$\mathcal{G} \textit{ validates } (\mathcal{C}, s(c)) \quad \textit{iff} \quad \mathcal{G}, c \models tr^+_{\emptyset,\mathcal{C}}(s).$$

The proof, given in the appendix, uses the approximation semantics of the μ-calculus, and the iterative computation of the well-founded semantics via the immediate consequence operator. In a nutshell, we show that for every a and i:

1. if $s(a)$ is in the i-th iteration of $W_{\mathcal{G},\mathcal{C}_s}(\emptyset)$, then there exists a j such that a is in the extension of the j-th approximation of $\mu X_s.\varphi = tr^+_{\emptyset,\mathcal{C}}(s)$, and conversely
2. if a is in the extension of the j-th approximation of $\mu X_s.\varphi = tr^+_{\emptyset,\mathcal{C}}(s)$, then for some j we have that $s(a)$ is in the i-th iteration of $W_{\mathcal{G},\mathcal{C}_s}(\emptyset)$.

Each of these claims is shown by induction on the shape of the formula, which means we also treat atoms of the form $s'(a)$ and $\neg s'(a)$ in the i-th iteration of $W_{\mathcal{G},\mathcal{C}_s}(\emptyset)$

$$
\begin{aligned}
||\Phi \wedge \psi||^{\mathcal{G}}_V &:= ||\Phi||^{\mathcal{G}}_V \cap ||\psi||^{\mathcal{G}}_V & ||A||^{\mathcal{G}}_V &:= A^{\mathcal{G}} \\
||\Phi \vee \psi||^{\mathcal{G}}_V &:= ||\Phi||^{\mathcal{G}}_V \cup ||\psi||^{\mathcal{G}}_V & ||\neg A||^{\mathcal{G}}_V &:= \Delta^{\mathcal{G}} \setminus A^{\mathcal{G}} \\
||[r]\Phi||^{\mathcal{G}}_V &:= \{e \in \Delta^{\mathcal{G}} \mid \text{ if } (e,e') \in r^{\mathcal{G}}, \text{ then } e' \in ||\Phi||^{\mathcal{G}}_V\} & ||a||^{\mathcal{G}}_V &:= a^{\mathcal{G}} \\
||\langle r\rangle\Phi||^{\mathcal{G}}_V &:= \{e \in \Delta^{\mathcal{G}} \mid \text{ there is } e' \text{ s.t. } (e,e') \in r^{\mathcal{G}}, e' \in ||\Phi||^{\mathcal{G}}_V\} & ||\neg a||^{\mathcal{G}}_V &:= \Delta^{\mathcal{G}} \setminus a^{\mathcal{G}} \\
||\mu X.\Phi||^{\mathcal{G}}_V &:= \bigcap\{U \subseteq \Delta^{\mathcal{G}} : ||\Phi||_{V[X\to U]} \subseteq U\} & ||X||^{\mathcal{G}}_V &:= V(X) \\
||\nu X.\Phi||^{\mathcal{G}}_V &:= \bigcup\{U \subseteq \Delta^{\mathcal{G}} : U \subseteq ||\Phi||_{V[X\to U]}\}
\end{aligned}
$$

Figure 3: Semantics of μ-calculus formulas.

$$
\begin{aligned}
tr^+_{S,\mathcal{C}}(s \wedge s') &:= tr^+_{S,\mathcal{C}}(s) \wedge tr^+_{S,\mathcal{C}}(s') & tr^-_{S,\mathcal{C}}(s \wedge s') &:= tr^-_{S,\mathcal{C}}(s) \vee tr^-_{S,\mathcal{C}}(s') \\
tr^+_{S,\mathcal{C}}(s \vee s') &:= tr^+_{S,\mathcal{C}}(s) \vee tr^+_{S,\mathcal{C}}(s') & tr^-_{S,\mathcal{C}}(s \vee s') &:= tr^-_{S,\mathcal{C}}(s) \wedge tr^-_{S,\mathcal{C}}(s') \\
tr^+_{S,\mathcal{C}}(\exists r.s) &:= \langle r\rangle tr^+_{S,\mathcal{C}}(s) & tr^-_{S,\mathcal{C}}(\exists r.s) &:= [r] tr^-_{S,\mathcal{C}}(s) \\
tr^+_{S,\mathcal{C}}(\forall r.s) &:= [r] tr^+_{S,\mathcal{C}}(s) & tr^-_{S,\mathcal{C}}(\forall r.s) &:= \langle r\rangle tr^-_{S,\mathcal{C}}(s) \\
tr^+_{S,\mathcal{C}}(A) &:= A & tr^-_{S,\mathcal{C}}(A) &:= \neg A \\
tr^+_{S,\mathcal{C}}(a) &:= a & tr^-_{S,\mathcal{C}}(a) &:= \neg a \\
tr^+_{S,\mathcal{C}}(\neg s) &:= tr^-_{S,\mathcal{C}}(s) & tr^-_{S,\mathcal{C}}(\neg s) &:= tr^+_{pos(S),\mathcal{C}}(s) \\
tr^+_{S,\mathcal{C}}(s) &:= \begin{cases} X_s & \text{if } s \in S \\ \mu X_s.tr^+_{S\cup\{s\},\mathcal{C}}(\mathcal{C}(s)) & \text{if } s \notin S \end{cases} & tr^-_{S,\mathcal{C}}(s) &:= \begin{cases} X_{\bar{s}} & \text{if } \bar{s} \in S \\ \nu X_{\bar{s}}.tr^-_{S\cup\{\bar{s}\},\mathcal{C}}(\mathcal{C}(s)) & \text{if } \bar{s} \notin S \end{cases}
\end{aligned}
$$

Figure 4: Translation functions, note the negated translation of negation, $tr^+_{S,\mathcal{C}}(\neg s)$: the well-founded semantics does not treat positive and negative information equally, which is captured by only keeping the positive shape names in the set S: $pos(S) := S \setminus \{\bar{s} \in S\}$.

and compare this with the extension of the subformulas $\mu X_{s'}.\psi$, resp. $\nu X_{\bar{s'}}.\psi$ in the j-approximation of $\mu X_s.\varphi = tr^+_{\emptyset,\mathcal{C}}(s)$. Note that the iterative computation of $W_{\mathcal{G},\mathcal{C}}$ and the interactive approximations of $tr^+_{\emptyset,\mathcal{C}}(s)$ operate quite differently and, in particular, they do not derive facts in the same number of iterations, hence the mismatch between the indices i and j.

This theorem gives us desired reduction from shape satisfiability to satisfiability in the full hybrid μ-calculus.

Corollary 1. *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be sets of shape constraints and s_1 and s_2 shape names. Then*

- *s is (finitely) satisfiable w.r.t. $\mathcal{C}_1$ under the well-founded semantics iff $tr^+_{\emptyset,\mathcal{C}_1}(s)$ is (finitely) satisfiable, and*
- *$(\mathcal{C}_1, s_1)$ (finitely) implies $(\mathcal{C}_2, s_2)$ under the well-founded semantics iff $tr^+_{\emptyset,\mathcal{C}_1}(s) \wedge \neg tr^+_{\emptyset,\mathcal{C}_2}$ is not (finitely) satisfiable.*

Implication and Satisfiability of Documents. We also reduce satisfiability and containment of SHACL documents to satisfiability in the full hybrid μ-calculus, as presented below. Here, we show how to deal with the (finite) document implication problem, and recall that in Proposition 1 we described how (finite) document satisfiability can be reduced to this case.

That is, for any document $(\mathcal{C}, \mathcal{T})$, let $\Theta_{\mathcal{C},\mathcal{T}}$ be the formula defined as a conjunction of the following formulas:

$$
\begin{aligned}
&\neg a \vee tr^+_{\emptyset,\mathcal{C}}(s) && \text{for all } (a,s) \in \mathcal{T} \\
&\neg A \vee tr^+_{\emptyset,\mathcal{C}}(s) && \text{for all } (A,s) \in \mathcal{T} \\
&\neg\langle r\rangle\top \vee tr^+_{\emptyset,\mathcal{C}}(s) && \text{for all } (r,s) \in \mathcal{T}.
\end{aligned}
$$

When evaluated on a graph node, $\Theta_{\mathcal{C},\mathcal{T}}$ tells us whether the node satisfies all required target shapes. To propagate $\Theta_{\mathcal{C},\mathcal{T}}$ to all (relevant) nodes of the graph, we use a greatest fixed point formula; for a document $(\mathcal{C}, \mathcal{T})$, and a role name p, let

$$
\Lambda_{\mathcal{C},\mathcal{T},p} = \nu X.(\Theta_{\mathcal{C},\mathcal{T}} \wedge \bigwedge_{r \in R \cup \{p\}} [r^-].X \wedge \bigwedge_{r \in R \cup \{p\}} [r].X),
$$

where R is the set of role names that appear in $(\mathcal{C}, \mathcal{T})$.

Theorem 4. *For each pair of SHACL documents $(\mathcal{C}, \mathcal{T})$ and $(\mathcal{C}', \mathcal{T}')$, a SHACL document $(\mathcal{C}, \mathcal{T})$ (finitely) implies $(\mathcal{C}', \mathcal{T}')$ iff*

$$
\bigwedge_{a \in I} \langle p\rangle(a \wedge \Lambda_{\mathcal{C},\mathcal{T},p}) \wedge \langle p\rangle\neg\Lambda_{\mathcal{C}',\mathcal{T}',p}
$$

is not (finitely) satisfiable, where I is the set of individuals that appear in $(\mathcal{C}, \mathcal{T})$ or $(\mathcal{C}', \mathcal{T}')$, and p a fresh role.

6 Complexity Results

Since the full hybrid μ-calculus is decidable, we obtain from Corollary 1 and Theorem 4 that the static analysis problems of interest are also decidable. This is particularly exciting in light of Theorem 1. However, we also want to obtain tight complexity results. The complexity of the μ-calculus and its variants is well understood, but, regrettably, our translation may incur an exponential blow-up due to the possible repetition of subformulas associated with each shape name. Fortunately, this blow-up affects only the representation of the translation as a single formula, and not the complexity of the problem.

In this section, we obtain complexity results for the static analysis problems under the well-founded semantics. The key insight is that the automata-based technique for the fully hybrid μ-calculus by Sattler and Vardi (2001) can be adapted to decide satisfiability of the formulas that result from our translation in time that is single exponential in the size of the original input, and not of the translated formulas. In a nutshell, instead of using all subformulas as states, we reuse the translation functions as presented in Figure 4 as the transition function, avoiding the problem of exponentially many states in the constructed automaton. We provide the full construction for the sake of completeness, but remark that the adaptation is relatively straightforward, as will be clear to the readers familiar with automata decision procedures for the μ-calculus.

Two-Way Alternating Parity Tree Automata. We recall 2ATA, which can move up and down possibly infinite trees with transitions that may move to combinations of states and directions. For $k \geq 1$, let $(\{1,\dots,k\}^*, \ell)$ be a k-ary Σ-*labelled tree* if ℓ maps each node $x \in \{1,\dots,k\}^*$ to its label $\ell(x) \in \Sigma$. For $1 \leq i \leq k$, we let $x \cdot i$ be the ith *child* of x. With $x \cdot 0$ we mean x itself, and $x \cdot -1$ indicates the *parent* of x; the node y such that $x = y \cdot i$ for some $1 \leq i \leq k$. Let $\mathcal{B}(I)$ be the set of *positive boolean formulas* over I, built inductively by applying $\wedge$ and $\vee$ starting from $\bot$, $\top$ and the elements of I. For a set $J \subseteq I$ and a formula $\varphi \in \mathcal{B}(I)$, we say that J satisfies φ iff assigning $\top$ to the elements in J and $\bot$ to those in $I \setminus J$, makes φ true. For a positive integer k, let $[k] = \{-1, 0, 1, \dots, k\}$. A *two-way alternating parity tree automaton* (2ATA) running over infinite tree encodings with branching degree k, is a tuple $\mathbf{A} = \langle \Sigma, Q, \delta, q_0, p \rangle$, where Σ is the input alphabet, Q is a finite set of states, $\delta : Q \times \Sigma \to \mathcal{B}([k] \times Q)$ is the transition function, $q_0 \in Q$ is the initial state and $p : Q \to \mathbb{N}$ is the priority function.

A run of a 2ATA $\mathbf{A}$ over a k-ary tree $(\{1,\dots,k\}^*, \ell)$, is a $(\{1,\dots,k\}^* \times Q)$-labelled tree $\mathcal{T}_r = (T_r, \ell_r)$ with $V_r \subseteq \{1,\dots,k\}^*$, satisfying:

1. $\epsilon \in V_r$ and $\ell_r(\epsilon) = (\epsilon, q_0)$.
2. Let $y \in V_r$ with $\ell_r(y) = (x, q)$ and $\delta(q, \ell(x)) = \varphi$. Then there is a (possibly empty) set $S = \{(c_1, q_1), \dots, (c_n, q_n)\} \subseteq [k] \times Q$ such that:
   - S satisfies φ, and
   - for all $1 \leq i \leq k$, we have that $y \cdot i \in V_r$ and $\ell_r(y \cdot i) = (x \cdot c_i, q_i)$.

Given an infinite path $p \subseteq V_r$, let $inf(p) \subseteq Q$ be the set of states that appear infinitely often in p. A run $\mathcal{T}_r$ of a 2ATA $\mathbf{A} = \langle \Sigma, Q, \delta, q_0, p \rangle$ is *accepting* if for every infinite path $p \subseteq V_r$, the state with the highest priority that occurs in $inf(p)$ is even.

Following Sattler and Vardi (2001), we define *guesses*, which fix the entire configuration of the individuals mentioned in the constraint set. We can enumerate all such configurations and build an automaton for each of them.

Definition 3. *A* guess $\mathbf{G} = (G, f, C)$ *for a constraint set $\mathcal{C}$ containing the nominals $a_1, \dots, a_l$ consists of a* guess list $G = (\gamma_1, \dots, \gamma_l)$*, a set of* connections $C \subseteq N_I \times N_R(\mathcal{C}) \times N_I$ *and a* guess mapping $f : \{1,\dots,l\} \to \{1,\dots,l\}$*, such that for each $1 \leq i, j \leq l$, $\emptyset \subsetneq \gamma_i \subseteq sub(\mathcal{C}) \cup \{\neg s \mid s \in sub(\mathcal{C})\}$ or $\gamma_i = \bot$, $a_i \in \gamma_{f(i)}$, $a_i \notin \gamma_j$, for all $j \neq f(i)$, if $N_I \cap \gamma_i = \emptyset$, then $\gamma_i = \bot$, and $(a_i, r, a_j) \in C$ implies $(a_j, r^-, a_i) \in C$.*

Proposition 4. *For each shapes pair $(\mathcal{C}, s)$ and each guess $\mathbf{G}$ for $\mathcal{C}$, we can define a 2ATA $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ such that:*

1. *If $tr^+_{\emptyset,\mathcal{C}}(s)$ is satisfiable, then for some guess $\mathbf{G}$, the language recognised by $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ is non-empty.*
2. *If for some guess $\mathbf{G}$ the language of $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ is not empty, then $tr^+_{\emptyset,\mathcal{C}}(s)$ is satisfiable.*
3. *The number of states in $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ is linear in $|sub(\mathcal{C})|$.*

The construction can be found in the appendix. It is a relatively straightforward adaptation of the translation in Sattler and Vardi (2001). The core of the construction uses two states $tr^+(\varphi)$ and $tr^-(\varphi)$ for each φ in $sub(\mathcal{C})$, corresponding to the two functions of the translation. Note that we can ignore the set S in this setting, as we no longer try to build a linear formula capturing a form of circularity. Moreover, the parity condition setting $p(tr^+(\varphi)) = 1$ for each φ in $sub(\mathcal{C})$, and for all other states $p(q) = 0$, reflects exactly the limited alternation of our translation.

(Finite) Shape Satisfiability w.r.t. Constraints. Based on the aforementioned 2ATA construction, we can now derive the first set of complexity results.

Theorem 5. *Deciding shape satisfiability w.r.t constraints for $\mathcal{ALCIO}$ SHACL is* EXPTIME*-complete, under the well-founded semantics.*

Proof. As noted by Sattler and Vardi (2001), emptiness of $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ can be decided in exponential time in the number of states. Moreover, the amount of guesses is also bounded by the amount of connections and guess lists, which means there are at most exponentially many automata $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ that need to be tested for emptiness, thus shape satisfiability w.r.t. constraints can be decided in EXPTIME.

EXPTIME-hardness follows from the fact that plain shape implication is equivalent to concept satisfiability in the description logic $\mathcal{ALCIO}$, which is known to be EXPTIME-complete (Tobies 2000). For a more detailed coverage of restricted settings that are already EXPTIME-hard, see (Oudshoorn 2025). □

For the finite version of the above problem we do not provide tight bounds for full $\mathcal{ALCIO}$ SHACL, but decidability already follows from finite satisfiability of full hybrid μ-calculus being decidable. Before discussing this, we first consider the finite model property under the well-founded semantics.

Proposition 5. *$\mathcal{ALCO}$ SHACL has the finite model property, $\mathcal{ALCI}$ SHACL does not, under the well-founded semantics.*

Proof. Given the result of Theorem 3, the finite model property for $\mathcal{ALCO}$ SHACL follows directly from (Tamura 2015). To see that $\mathcal{ALCI}$ SHACL does not have the finite model property, consider the following set of (stratified) constraints: $\mathcal{C} = \{s \leftarrow \neg s', s' \leftarrow \forall r.s' \wedge \neg s'', s'' \leftarrow \forall r^-.s''\}$. Satisfiability of s against $\mathcal{C}$ corresponds to satisfiability of $cln(tr^+_{\emptyset,\mathcal{C}}(s)) = \nu X_{\bar{s'}}.\langle r\rangle X_{\bar{s'}} \wedge \mu Y_{s''}.[r^-]Y_{s''}$, which is a well-known example of a modal μ-calculus formula that only has infinite models. □

Theorem 6. *Deciding finite shape satisfiability w.r.t. constraints for $\mathcal{ALCIO}$ SHACL is decidable, under the well-founded semantics.*

Proof. Given the result of Theorem 3, the finite satisfiability of the hybrid μ-calculus being decidable suffices. The latter follows from the finite satisfiability of guarded fixed point logic, which naturally extends the modal μ-calculus with backwards modalities (Bárány and Bojanczyk 2012). In guarded fixed point, given that there is no bound on the arity of predicates, we can add a prefix $x_1, \ldots, x_n$ to every predicate, where n is the amount of individuals used, and use x_i in place of the i-th individual, to model the usage of nominals (Ten Cate and Franceschet 2005). □

For upper bounds, recall the translation of SHACL into μ-calculus may come at the cost of a formula of size exponential in the size of the constraint set. Thus, we can combine finite satisfiability results for different fragments of the full hybrid μ-calculus with the result of our translation (Theorem 3) by adding one exponential. Considering how nominals can be eliminated in guarded formulas as described above, at the cost of not being able to bound the arity of predicates (Ten Cate and Franceschet 2005), it follows from the work by Bárány and Bojanczyk (2012) that finite satisfiability for $\mathcal{ALCIO}$ SHACL can be solved in 3-ExpTime. For $\mathcal{ALCI}$ SHACL, 2-ExpTime-membership for deciding finite satisfiability already follows from work by Bojanczyk (2002). We have no reason to believe these upper bounds are tight: they are obtained via multiple translations that cause one or more exponential blow-ups, which could well be avoidable in a more direct approach.

(Finite) Satisfiability and Implication of Documents. With a small update of the automaton construction, the ExpTime complexity results transfer to the problem of document satisfiability.

Theorem 7. *Deciding document satisfiability for $\mathcal{ALCIO}$ SHACL is* ExpTime*-complete, under the well-founded semantics. Finite document satisfiability for $\mathcal{ALCIO}$ SHACL is decidable.*

A direct consequence of this result is that for $\mathcal{ALCO}$ SHACL deciding finite satisfiability w.r.t. constraints under the well-founded semantics is ExpTime-complete, since the fragment has the finite model property; see Proposition 5.

As the complement of an 2ATA can be constructed in polynomial time (Muller and Schupp 1987), it follows that deciding whether $(\mathcal{C}, \mathcal{T})$ implies $(\mathcal{C}', \mathcal{T}')$ under the well-founded semantics for $\mathcal{ALCIO}$ SHACL can be decided by taking the conjunction of the automaton checking document satisfiability for $(\mathcal{C}, \mathcal{T})$ and the complement of the automaton recognising document satisfiability for $(\mathcal{C}', \mathcal{T}')$.

Theorem 8. *Deciding document implication for $\mathcal{ALCIO}$ SHACL is* ExpTime*-complete, under the well-founded semantics. Finite document implication for $\mathcal{ALCIO}$ SHACL is decidable.*

Least Fixed Point Semantics. Apart from the supported, stable model and well-founded semantics for recursive SHACL, another semantics proposed in the literature is the least fixed point semantics for stratified SHACL (Ahmetaj et al. 2023; Oudshoorn, Ortiz, and Šimkus 2026). As the name already suggests, this is a semantics only using a least fixed point in its definition, making it conceptually easier. Nonetheless, this suffices to compute either the well-founded or stable model semantics over stratified constraints; if $\mathcal{C}$ is stratified, these three semantics coincide. In particular, this means the discussed static analysis problems for least fixed points semantics can be simply copied from the well-founded setting discussed above. Moreover, the translation of stratified SHACL, and thus also of SHACL under the least fixed point semantics, would fall in the alternation free μ-calculus. To see this, in a nutshell, note that in a μ-calculus formula with alternation like $\mu X_s.\varphi$ such that $\nu X_{\bar{s'}}.\varphi(X_s) \in sub(\varphi)$ we find s' depending negatively on s; because of the ν-formula appearing inside the μ-formula, and s depending negatively on s'; because of X_s appearing inside the ν-subformula. Clearly, this makes the constraint set unstratified. We are optimistic that this can be exploited to achieve more scalable static analysis results.

7 Conclusion and Discussion

In this paper, by establishing a connection to the full hybrid μ-calculus, we provide the first decidability and complexity results for basic static analysis problems in recursive SHACL under the well-founded semantics. These results contrast sharply with the undecidability results we show for the supported and stable model semantics. A key observation is that the implication problem becomes computationally unmanageable when the underlying semantics can assign multiple shape assignments to a graph. The well-founded semantics yields a unique shape assignment for every graph, which forms one of the bases under our decidability result.

There are several directions for future work. First, in the context of practical applications, it is relevant to understand static analysis in the presence of a richer syntax for shape expressions. For instance, adding numeric restrictions (such as qualified number restrictions in DLs) to SHACL under the well-founded semantics is a natural next step. This requires understanding how a well-founded semantics can be defined in this setting and determining how our algorithmic methods can be adapted to it. Second, our translation from SHACL under the well-founded semantics into the full hybrid μ-calculus raises the question of whether a similar translation exists for Datalog with negation under the well-founded semantics into Fixpoint Logic. A natural follow-up question is whether decidability results for Guarded Fixpoint Logic can yield new positive results for restricted Datalog fragments under the well-founded semantics.

Acknowledgments

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101034440.

In addition, this work was partially funded by the Austrian Science Fund (FWF) projects P30873, PIN8884924, and 10.55776/COE12.

References

Ahmetaj, S.; Ortiz, M.; Oudshoorn, A.; and Šimkus, M. 2023. Reconciling SHACL and ontologies: Semantics and validation via rewriting. In Gal, K.; Nowé, A.; Nalepa, G. J.; Fairstein, R.; and Radulescu, R., eds., *ECAI 2023 - 26th European Conference on Artificial Intelligence*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, 27–35. IOS Press.

Ahmetaj, S.; Boneva, I.; Hidders, J.; Jakubowski, M.; Labra-Gayo, J.-E.; Martens, W.; Mogavero, F.; Murlak, F.; Okulmus, C.; Savković, O.; Šimkus, M.; and Tomaszuk, D. 2026. Common foundations for recursive shape languages. In *Proc. of 23rd International Conference on Principles of Knowledge Representation and Reasoning (KR 2026)*.

Andresel, M.; Corman, J.; Ortiz, M.; Reutter, J. L.; Savkovic, O.; and Šimkus, M. 2020. Stable model semantics for recursive SHACL. In Huang, Y.; King, I.; Liu, T.; and van Steen, M., eds., *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, 1570–1580. ACM / IW3C2.

Apt, K. R.; Blair, H. A.; and Walker, A. 1988. Towards a theory of declarative knowledge. In Minker, J., ed., *Foundations of Deductive Databases and Logic Programming*. Morgan Kaufmann. 89–148.

Arnold, A., and Niwinski, D. 2001. Rudiments of μ-calculus. *Studies in Logic and the Foundations of Mathematics* 146.

Bárány, V., and Bojanczyk, M. 2012. Finite satisfiability for guarded fixpoint logic. *Inf. Process. Lett.* 112(10):371–375.

Berger, R. 1966. *The undecidability of the domino problem*. Number 66. American Mathematical Soc.

Bogaerts, B.; Jakubowski, M.; and Van den Bussche, J. 2022. SHACL: A description logic in disguise. In Gottlob, G.; Inclezan, D.; and Maratea, M., 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*, 75–88. Springer.

Bojanczyk, M. 2002. Two-way alternating automata and finite models. In Widmayer, P.; Ruiz, F. T.; Bueno, R. M.; Hennessy, M.; Eidenbenz, S. J.; and Conejo, R., eds., *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, volume 2380 of *Lecture Notes in Computer Science*, 833–844. Springer.

Bonatti, P. A.; Lutz, C.; Murano, A.; and Vardi, M. Y. 2008. The complexity of enriched mu-calculi. *Log. Methods Comput. Sci.* 4(3).

Corman, J.; Reutter, J. L.; and Savkovic, O. 2018. Semantics and validation of recursive SHACL. In Vrandecic, D.; Bontcheva, K.; Suárez-Figueroa, M. C.; Presutti, V.; Celino, I.; Sabou, M.; Kaffee, L.; and Simperl, E., 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*, 318–336. Springer.

Di Stefano, F., and Šimkus, M. 2024. Stable model semantics for description logic terminologies. In Wooldridge, M. J.; Dy, J. G.; and Natarajan, S., eds., *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, 10484–10492. AAAI Press.

Oudshoorn, A.; Ortiz, M.; and Šimkus, M. 2024. Reasoning with the core chase: the case of SHACL validation over ELHI knowledge bases. In Giordano, L.; Jung, J. C.; and Ozaki, A., 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.

Oudshoorn, A.; Ortiz, M.; and Šimkus, M. 2026. SHACL validation in the presence of ontologies: Semantics and rewriting techniques. *Artificial Intelligence* 352:104483.

Oudshoorn, A. 2025. SHACL satisfiability: What can we learn from dls? In *Proceedings of the 38th International Workshop on Description Logics (DL 2025), Opole, Poland, September 3-6, 2025*, volume 4091 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Knublauch, H., and Kontokostas, D. 2017. Shapes Constraint Language (SHACL). W3C Recommendation, W3C. https://www.w3.org/TR/shacl/.

Leinberger, M.; Seifer, P.; Rienstra, T.; Lämmel, R.; and Staab, S. 2020. Deciding SHACL shape containment through description logics reasoning. In Pan, J. Z.; Tamma, V.; d'Amato, C.; Janowicz, K.; Fu, B.; Polleres, A.; Seneviratne, O.; and Kagal, L., 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*, 366–383. Springer.

Muller, D. E., and Schupp, P. E. 1987. Alternating automata on infinite trees. *Theor. Comput. Sci.* 54:267–276.

Okulmus, C., and Šimkus, M. 2024. SHACL validation under the well-founded semantics. In Marquis, P.; Ortiz, M.; and Pagnucco, M., eds., *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024*.

Ortiz, M. 2023. A short introduction to SHACL for logicians. In Hansen, H. H.; Scedrov, A.; and de Queiroz, R. J. G. B., eds., *Logic, Language, Information, and Computation - 29th International Workshop, WoLLIC 2023, Halifax, NS, Canada, July 11-14, 2023, Proceedings*, volume 13923 of *Lecture Notes in Computer Science*, 19–32. Springer.

Pareti, P.; Konstantinidis, G.; and Mogavero, F. 2022. Satisfiability and containment of recursive SHACL. *J. Web Semant.* 74:100721.

Sattler, U., and Vardi, M. Y. 2001. The hybrid μ-calculus. In Goré, R.; Leitsch, A.; and Nipkow, T., eds., *Automated Reasoning, First International Joint Conference, IJCAR 2001, Siena, Italy, June 18-23, 2001, Proceedings*, volume 2083 of *Lecture Notes in Computer Science*, 76–91. Springer.

Tamura, K. 2015. A small model theorem for the hybrid μ-calculus. *J. Log. Comput.* 25(2):405–441.

Ten Cate, B., and Franceschet, M. 2005. Guarded fragments with constants. *J. Log. Lang. Inf.* 14(3):281–288.

Tobies, S. 2000. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *J. Artif. Intell. Res.* 12:199–217.

Van Gelder, A.; Ross, K. A.; and Schlipf, J. S. 1991. The well-founded semantics for general logic programs. *J. ACM* 38(3):620–650.

Van Gelder, A. 1993. The alternating fixpoint of logic programs with negation. *J. Comput. Syst. Sci.* 47(1):185–221.

A Proofs for Section 4 (Full Hybrid μ-calculus)

Every formula can be *cleaned*, as defined recursively by the function cln, given in Figure 5, with the main advantage of improving the readability of the formula.

$$
\begin{aligned}
cln(\Phi \wedge \Phi') &:= cln(\Phi) \wedge cln(\Phi') \\
cln(\Phi \vee \Phi') &:= cln(\Phi) \vee cln(\Phi') \\
cln(\langle r \rangle \Phi) &:= \langle r \rangle cln(\Phi) \qquad cln(A) := A \\
cln([r]\Phi) &:= [r] cln(\Phi) \qquad cln(a) := a \\
cln(\neg \Phi) &:= \neg cln(\Phi) \qquad cln(X) := X \\
cln(\sigma X.\Phi) &:= \begin{cases} cln(\Phi) & \text{if } X \in sub(\Phi) \\ \sigma X.cln(\Phi) & \text{if } X \notin sub(\Phi) \end{cases}
\end{aligned}
$$

Figure 5: Cleaning function.

B Proofs for Section 5 (Translating SHACL into μ-calculus)

Note that the constraint definitions that are not reachable from a target are irrelevant to its validation, and hence to test s we can just use $\mathcal{C}_s$, defined in the following way.

Definition 4. *Given a constraint set $\mathcal{C}$, let $cl_{\mathcal{C}}(s)$, the closure of a shape name s w.r.t. $\mathcal{C}$, be defined as the smallest set of shape names $s' \in N_S$ such that $s \in cl_{\mathcal{C}}(s)$, and furthermore, if $s' \in cl_{\mathcal{C}}(s)$ and $s' \leftarrow \varphi \in \mathcal{C}$ such that $s'' \in sub(\varphi)$, then $s'' \in cl_{\mathcal{C}}(s)$. Let the restriction of a constraint set $\mathcal{C}$ to a shape name s be the following constraint set:*

$$\mathcal{C}_s := \{s' \leftarrow \varphi \in \mathcal{C} \mid s' \in cl_{\mathcal{C}}(s)\}.$$

We call these constraints reachable from s.

Recall that also $s' \in sub(\neg s')$. It is straightforward the following must hold.

Lemma 1. *For each constraint set $\mathcal{C}$ and interpretation $\mathcal{G}$, we have $\mathcal{G}$ validates $(\mathcal{C}, s(a))$ iff $\mathcal{G}$ validates $(\mathcal{C}_s, s(a))$.*

Without loss of generality, the proofs below assume that all constraints in $\mathcal{C}$ are reachable from s. We first establish an assisting lemma on the structure of the constructed modal μ-calculus formulas.

Lemma 2. *Given $\nu X.\varphi \in sub(tr^{+}_{\emptyset,\mathcal{C}}(s))$ such that $X \in sub(\varphi)$, then there is no $\mu X.\psi \in sub(\nu X.\varphi)$ such that $X \in sub(\psi)$.*

Assuming the contrary quickly leads to the conclusion that in that case X would not have been introduced; as only the positive shape names remain in the rule $tr^{-}_{S,\mathcal{C}}(\neg s) := tr^{+}_{pos(S),\mathcal{C}}(s)$.

Given an interpretation $\mathcal{G}$, let $atoms_{\mathcal{G}}$ be a (partial) function that maps (approximations of) modal mu-calculus formulas to sets of atoms, defined in the following way

$$
\begin{aligned}
atoms_{\mathcal{G}}(\mu^i X_s.\varphi) := &\{s(a) \mid a \in ||\mu^i X_s.\varphi||^{\mathcal{G}}_V\} \\
&\cup \{s'(a) \mid \mu X_{s'}.\psi \in sub(\varphi), a \in ||\mu X_{s'}.\psi||^{\mathcal{G}}_V\} \\
&\cup \{\neg s'(a) \mid \nu X_{\bar{s'}}.\psi \in sub(\varphi), a \in ||\nu X_{\bar{s'}}.\psi||^{\mathcal{G}}_V\}.
\end{aligned}
$$

We use the notation S_j as a shorthand for $W^{\uparrow j}_{\mathcal{G},\mathcal{C}}(\emptyset)$, whenever $\mathcal{G}$ and $\mathcal{C}$ are understood from the context.

Proposition 6. *For each constraint set $\mathcal{C}$, each shape name s, each interpretation $\mathcal{G}$ and each natural number i, if $t'(b) \in W^{\uparrow i}_{\mathcal{G},\mathcal{C}_s}(\emptyset)$, for $t' = s'$, or $t' = \neg s'$, then there exists a j such that $t'(b) \in atoms_{\mathcal{G}}(\mu^j X_s.\varphi)$, for $\mu X_s.\varphi = tr^{+}_{\emptyset,\mathcal{C}}(s)$.*

Proof. Note that because we consider $\mathcal{C}_s$, there exists a subformula of the form $\{\mu X_{s''}.\psi, \nu X_{\bar{s''}}.\psi\} \cap sub(tr^{+}_{\emptyset,\mathcal{C}}(s)) \neq \emptyset$ for each $s'' \leftarrow \varphi \in \mathcal{C}_s$. Moreover, as the function $tr^{+}_{\emptyset,\mathcal{C}}$ is alternating between the positive (only introducing subformulas of the form $\mu X.\psi$) and negative version, $tr^{-}_{S,\mathcal{C}}$ (only introducing $\nu X.\psi$-subformulas) exactly when some $s \leftarrow \neg s'$ shows up, we must also conclude that if $s''(b) \in S_i$ for some i, there must be $\mu X_{s''}.\psi \in sub(tr^{+}_{\emptyset,\mathcal{C}}(s))$, and similarly with $\nu X_{\bar{s''}}$ for $\neg s''(b) \in \mathcal{S}_i$. We prove this proposition by induction on i. For the induction basis, note $W^{\uparrow 0}_{\mathcal{G},\mathcal{C}_s}(\emptyset) = S_0 = \emptyset$. For the induction step, assume there exists some $t(b) \in S_i \setminus S_{i-1}$. We distinguish two cases: (i) $t = s'$ and (ii) $t = \neg s'$.

(i) Let $s' \leftarrow \varphi \in \mathcal{C}$ be the unique constraint with head s'. We distinguish the following cases.

- $\varphi = A$. Note that $s'(b) \in S_i$ implies that $b \in A^{\mathcal{G}}$. Furthermore, we find there exists some $\mu X_{s'}.\psi \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$. Note that $\psi = tr^+_{S\cup s',\mathcal{C}_s}(A) = A$, independent of what is contained in S. As $||\mu X'_s.A||^{\mathcal{G}}_V = A^{\mathcal{G}}$, we find $s'(b) \in atoms_{\mathcal{G}}(\mu^j X_s.\varphi)$ for each $j \geq 1$.
- $\varphi = \neg s''$. Note that if $s'(b) \in W^{\uparrow i}_{\mathcal{G},\mathcal{C}_s}(\emptyset)$, then $\neg s''(b) \in W^{\uparrow i-1}_{\mathcal{G},\mathcal{C}_s}(\emptyset)$, thus, by using the induction hypothesis, we find that there exists a j such that $\neg s''(b) \in atoms_{\mathcal{G}}(\mu^j X_s.\varphi)$. Following the definition of $atoms_{\mathcal{G}}$, this means there must be a $\nu X_{s''}.\psi \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$ such that $b \in ||\nu X_{\bar{s''}}.\psi||^{\mathcal{G}}_{V_j}$, where V_j is such that $V_j(X_s) = ||\mu^{j-1} X_s.\varphi||^{\mathcal{G}}_V$. Considering the definition of $tr^+_{\emptyset,\mathcal{C}}(s)$, we distinguish two options: (a) either we also find the subformula $\mu X_{s'}.\nu X_{\bar{s''}}.\psi \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$, or (b), only the subformula $\mu X_{s'}.X_{\bar{s''}} \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$. As (b) is in contradiction with Lemma 2, we only consider case (a). If $X_{s'} \in sub(\psi)$, this means there is some subformula $\mu X_{s'}.\chi(X_{s'}) \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$ for which we can use a similar proof strategy using the least-fixed point approximation. If $X_{s'} \notin sub(\psi)$, this means that $||\mu X_{s'}.\nu X_{\bar{s''}}.\psi||^{\mathcal{G}}_{V_j} = ||\nu X_{\bar{s''}}.\psi||^{\mathcal{G}}_{V_j}$, and thus $s'(b) \in atoms_{\mathcal{G}}(\mu^j X_s.\varphi)$.
- Other cases are treated similarly.

(ii) Let U be the greatest unfounded set w.r.t. S_{i-1}, $\mathcal{G}$ and $\mathcal{C}_s$, we distinguish four cases: (a) $s'(b) \in U$ such that $s' \leftarrow A \in \mathcal{C}$ is the unique constraint with head s'; (b) $s'(b) \in U$ such that $s' \leftarrow \neg s'' \in \mathcal{C}$ is the unique constraint with head s'; (c) $s'(b) \in U$ such that $s' \leftarrow \exists r.s'' \in \mathcal{C}$ is the unique constraint with head s' and $\{e \in \Delta^{\mathcal{G}} \mid (b,e) \in r^{\mathcal{G}}\} = \emptyset$; and (d) all $s'(b) \in U$ that do not fit in any of the previous categories. Note that cases (a) and (c) can be treated similarly to the cases $\varphi = A$, resp. $\varphi = \forall r.s''$ in part (i). Thus, we will only address cases (b) and (d) here.

(b) $\varphi = \neg s''$. Note that $\lceil\varphi\rceil^{\mathcal{G}}_{S_{i-1}\cup\neg.U} = \{a \in \Delta^{\mathcal{G}} \mid s''(a) \notin S_{i-1} \cup \neg.U\}$. Thus, $b \notin \lceil\varphi\rceil^{\mathcal{G}}_{S_{i-1}\cup\neg.U}$ implies that $s''(b) \in S_{i-1}$. Now, we may use the induction hypothesis to conclude there exists some j such that $s''(b) \in atoms_{\mathcal{G}}(tr^+_{\emptyset,\mathcal{C}}(s))$. The rest of the argument can be treated similar to the cases in (i).

(d) Take some $s'(b) \in U$ and note that there must exist some $\nu X_{\bar{s'}}.\psi \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$. Now let $L = \{a \in \Delta^{\mathcal{G}} \mid s'(a) \in U\}$, we aim to show that $L \subseteq ||\nu X_{\bar{s'}}.\psi||^{\mathcal{G}}_{V[X_{\bar{s'}} \mapsto L]}$. We distinguish two cases: (α) either $X_{\bar{s'}} \in sub(\psi)$, or ($\beta$) not. In the latter case, we may distinguish two settings again, either there exists some $\nu X_{\bar{s''}}.\psi' \in sub(tr^+_{\emptyset,\mathcal{C}}(s))$ such that $\nu X_{\bar{s'}}.\psi \in sub(\psi')$ and $X_{\bar{s''}} \in sub(\psi)$, that is, $\nu X_{\bar{s'}}.\psi$ appears within some "loop" of negated atoms, or not. The case of $\nu X_{\bar{s'}}.\psi$ in the "loop" will follow from the way case (α) is considered, whereas the outside-of-the-loop-case will either fall directly in cases (a)-(c), or can be brought back to cases of the form (a)-(c) with techniques similar to the ones in (i).
That is, we solely focus on case (α) here; if $X_{\bar{s'}} \in sub(\psi)$, there exist one or more chains of axioms $C_i = \{s_{i_1} \leftarrow \varphi_{i_1}, \ldots, s_{i_n} \leftarrow \varphi_{i_n}\}$ such that $s_{i_1} = s'$, each φ_{i_j} is of one of the forms $s'' \wedge s'''$, $s'' \vee s'''$, $\exists r.s''$ or $\forall r.s''$, for each $1 \leq j \leq n-1$ we have $s_{i_{j+1}} \in sub(\varphi_{i_j})$, and lastly, we find $s_{i_1} \in sub(\varphi_{i_n})$. To avoid redundancy, we also assume $s' \neq s_{i_j}$ for each $1 < j \leq n$. Note that by combining the insights of Lemma 2 with the definition of the translation function, we find that for each $1 < j \leq n$, there exists some $\nu X_{\bar{s_{i_j}}}.\psi_{i_j} \in sub(\psi)$ such that $\nu X_{\bar{s_{i_j}}}.\psi_{i_j} \in sub(\psi_{i_{j-1}})$.
Now let U' be the greatest subset of U such that if $s(a) \in U'$, then there exists some chain of axioms C_i such that $s = s_{i_j}$ for some $1 \leq j \leq n$, and moreover, for all $1 \leq j \leq n$, if $s_{i_j}(a) \in U'$ and $s_{i_j} \leftarrow \varphi_{i_j} \in \mathcal{C}$, then
* if $\varphi_{i_j} \in \{s_{i_{j+1}} \wedge s', s' \wedge s_{i_{j+1}}, s_{i_{j+1}} \vee s', s' \vee s_{i_{j+1}}\}$, then $s_{i_{j+1}}(a) \in U'$; and
* if $\varphi_{i_j} \in \{\exists r.s_{i_{j+1}}, \forall r.s_{i_{j+1}}\}$, then there exists some $b \in \{b \in \Delta^{\mathcal{G}} \mid (a,b) \in r^{\mathcal{G}}\}$ such that $s_{i_{j+1}}(b) \in U'$,
where $s_{i_{n+1}}$ is used as a substitute name for s_{i_1}.
For each chain of axioms C_i, let $L_{i_j} = \{a \in \Delta^{\mathcal{G}} \mid s_{i_j}(a) \in U'\}$. As $s_{i_1} = s'$ for each chain of axioms C_i, we find $L_{i_1} = L_{k_1}$ for each pair of axiom chains C_i and C_k. We want to show that

$$L_{i_j} \subseteq ||\nu X_{\bar{s_{i_j}}}.\psi_{i_j}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]} \tag{2}$$

for each constraint $s_{i_j} \leftarrow \varphi_{i_j}$ appearing in one of the axiom chains C_i, assuming that for each $s_{i_j} \leftarrow \varphi$ with $\varphi \in \{s_{i_{j+1}} \vee s', s' \vee s_{i_{j+1}}\}$, $\{a \in \Delta^{\mathcal{G}} \mid s_{i_j}(a) \in U'\} \subseteq ||\nu X_{\bar{s'}}.\psi'||^{\mathcal{G}}_V$, where $V(X_{\bar{s_{k_1}}}) = L_{k_1}$ in case $s' = s_{k_l}$, that is, s' appears in some other chain, or where V is in the state it is in after approximating the previous layer in case s' does not appear in any other chain. To see (2) holds, note it is immediate that $L_{i_1} \subseteq ||X_{\bar{s_{i_1}}}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]}$. Now assume $L_{i_j} \subseteq ||\nu X_{\bar{s_{i_j}}}.\psi_{i_j}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]}$, and let $s_{i_{j-1}}(b) \in L_{i_{j-1}}$. We distinguish four cases: conjunction, disjunction, and existential and universal quantification. To illustrate the main idea, we treat the conjunction case.
* Let $s_{i_{j-1}} \leftarrow s_{i_j} \wedge s' \in C_i$ (or similarly $s_{i_{j-1}} \leftarrow s' \wedge s_{i_j} \in C_i$). In this case $\psi_{i_{j-1}} = \nu X_{\bar{s_{i_j}}}.\psi_{i_j} \vee \chi$, for some χ representing the ν formula for s' not included in the discussed loop, and thus $||\nu X_{\bar{s_{i_j}}}.\psi_{i_j}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]} \cup ||\chi||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]} = ||\psi_{i_{j-1}}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]}$. As $X_{\bar{s_{i_{j-1}}}} \notin sub(\psi_{i_{j-1}})$, we find $||\psi_{i_{j-1}}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]} = ||\nu X_{\bar{s_{i_{j-1}}}}.\psi_{i_{j-1}}||^{\mathcal{G}}_{V[X_{\bar{s_{i_1}}} \mapsto L_{i_1}]}$.

Moreover, we find for each $a \in L_{i_{j-1}}$, we have $a \in L_{i_j}$, or $a \in ||\chi||^{\mathcal{G}}_{V[X_{s_{\bar{i}_1}} \mapsto L_{i_1}]}$; suppose $a \in L_{i_{j-1}}$, then $a \notin \lceil s_{i_j} \wedge s' \rceil^{\mathcal{G}}_{S \cup \neg.U'}$, which means $a \notin \lceil s_{i_j} \rceil^{\mathcal{G}}_{S \cup \neg.U'}$ or $\lceil s_{i_j} \wedge s' \rceil^{\mathcal{G}}_{S \cup \neg.U'}$. In the first case, note that $\lceil s_{i_j} \rceil^{\mathcal{G}}_{S \cup \neg.U'} = \{c \in \Delta^{\mathcal{G}} \mid \neg s_{i_j} \notin S \cup \neg.U'\}$, thus $s_{i_j}(a) \notin U'$, which means that $a \in L_{i_j}$. In the second case, $a \notin \lceil s' \rceil^{\mathcal{G}}_{S \cup \neg.U'}$, there are again two options: either $s' = s_{k_l}$, in which case a similar argument for the loop k suffices to show $a \in L_{k_l} \subseteq ||\chi||^{\mathcal{G}}_{V[X_{s_{\bar{i}_1}} \mapsto L_{i_1}]}$; or s' does not appear in any loop. In particular, this means there are no atoms of the form $s'(b) \in U'$, by definition of U'. If we now look at $a \notin \lceil s' \rceil^{\mathcal{G}}_{S \cup \neg.U'}$, this implies $s'(a) \in U'$, or $\neg s'(a) \in S$. Thus, $\neg s'(a) \in S$. So, we are back in a non-loop case and can use the reasoning of before to conclude: $a \in ||\chi||^{\mathcal{G}}_{V[X_{s_{\bar{i}_1}} \mapsto L_{i_1}]}$. Combining all subsumptions and equalities leads to the conclusion that $L_{i_{j-1}} \subseteq ||\nu X_{s_{i_{j-1}}}.\psi_{i_{j-1}}||^{\mathcal{G}}_{V[X_{s_{\bar{i}_1}} \mapsto L_{i_1}]}$, as required.

□

Proposition 7. *For each constraint set $\mathcal{C}$, each interpretation $\mathcal{G}$ and each natural number i, such that $\mu X_s.\varphi = tr^+_{\emptyset,\mathcal{C}}(s)$ if $s'(b) \in atoms_{\mathcal{G}}(\mu^i X_s.\varphi)$, then there exists j such that $s'(b) \in W^{\uparrow j}_{\mathcal{G},\mathcal{C}}(\emptyset)$.*

Proof. As we are considering a least fixed point, it suffices to prove the following claim.

Claim 1. *Given some $\sigma X_t.\varphi$ and a valuation function V, and assume that for each variable $X_{t'} \in sub(\varphi)$ that is free in $\sigma X_t.\varphi$, we have, if $b \in V(X_{t'})$, then there exists a j such that $b \in \lfloor t' \rfloor^{\mathcal{G}}_{S_j}$. In that case, there exists a j' such that $||\sigma X_t.\varphi||^{\mathcal{G}}_V \subseteq \lfloor t \rfloor^{\mathcal{G}}_{S_{j'}}$.*

We prove the above claim by induction on the construction of φ. We distinguish three cases: (i) $\sigma X.\varphi$ such that $X \notin sub(\varphi)$, and in case $X \in sub(\varphi)$: (ii) $\sigma = \mu$ and (iii) $\sigma = \nu$.

(i) - $\varphi = A$. Clearly $||\sigma X_t.A||^{\mathcal{G}}_V = ||A||^{\mathcal{G}}_V = A^{\mathcal{G}} = \lfloor A \rfloor^{\mathcal{G}}_{S_j}$, as $\sigma X_t.A$ can only be constructed in case $t \leftarrow A \in \mathcal{C}$, we find that if $b \in ||\sigma X_t.A||^{\mathcal{G}}_V$, then $b \in \lfloor t \rfloor^{\mathcal{G}}_{S_{j+1}}$.

- $\sigma X_t.\varphi = \mu X_t.\nu X_{\bar{s}}.\psi$. As $X_t \notin sub(\psi)$, we find $||\mu X_t.\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V = ||\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V$, then by IH, there exists some j such that $||\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V \subseteq \lfloor \neg s \rfloor^{\mathcal{G}}_{S_j}$. As $\mu X_t.\nu X_{\bar{s}}.\psi$ will only be constructed given some $t \leftarrow \neg s \in \mathcal{C}$, we find that if $b \in ||\mu X_t.\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V$, then $b \in \lfloor t \rfloor^{\mathcal{G}}_{S_{j+1}}$.

- Other cases are treated similarly.

(ii) - $\mu X_t.\varphi(X_t) = \mu X_t.\sigma X_s.\psi(X_t)$. In this case, we find $||\mu X_t.\sigma X_s.\psi(X_t)||^{\mathcal{G}}_V$ can be determined by using the approximation function $||\mu^0 X_t.\sigma X_s.\psi(X_t)||^{\mathcal{G}}_V = \emptyset$ and $||\mu^{i+1} X_t.\sigma X_s.\psi(X_t)||^{\mathcal{G}}_V = ||\sigma X_s.\psi(X_t)||^{\mathcal{G}}_{V[X_t \mapsto ||\mu^i X_t.\sigma X_s.\psi(X_t)||^{\mathcal{G}}_V]}$. Note that for the i-th approximated case, we can derive that there exists some j such that $||\mu^i X_t.\sigma X_s.\psi(X_t)||^{\mathcal{G}}_V \subseteq \lfloor t \rfloor^{\mathcal{G}}_{S_j}$ by a repetitive usage of the induction hypothesis. As we are considering a least fixed point, this eventually suffices.

- Other cases are treated similarly.

(iii) $\nu X_t.\varphi(X_t)$. Following Lemma 2, all $\sigma X_{t'}.\psi \in sub(\varphi)$ such that $X_t \in sub(\psi)$ are such that $\sigma = \nu$. We collect all relevant shape names in the following set: let $W = \{s \in N_S \mid \nu X_{\bar{s}}.\psi \in sub(\varphi), X_t \in sub(\psi)\}$. Note that for all $\nu X_{\bar{s}}.\psi \in sub(\varphi)$ such that $X_t \notin sub(\psi)$ the induction hypothesis may be applied to derive that there exists some j such that $||\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V \subseteq \lfloor \neg s \rfloor^{\mathcal{G}}_{S_j}$. Let j' be the maximum of all these j. Now let $U = \{s(a) \mid a \in ||\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V, \nu X_{\bar{s}}.\psi \in sub(\nu X_t.\varphi), s \in W\}$.
In what follows, we will show U is an unfounded set w.r.t. $S_{j'}$, $\mathcal{G}$ and $\mathcal{C}$. To this end, we have to show that for all $s(a) \in U$ we find $a \notin \lceil \mathcal{C}(s) \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}$. Note that if $s \leftarrow \varphi' \in \mathcal{C}$ such that $s \in W$, then φ' is of the form $\exists r.s'$, $\forall r.s'$, $s \leftarrow s' \wedge s''$ or $s \leftarrow s' \vee s''$. Thus, we may consider $\mathcal{C}(s) \in \{\exists r.s', \forall r.s', s' \wedge s'', s' \vee s''\}$:

- $\mathcal{C}(s) = \forall r.s'$. In this case, the subformula looks like $\nu X_{\bar{s}}.\langle r \rangle \nu X_{\bar{s'}}.\psi'$, or $\nu X_{\bar{s}}.\langle r \rangle X_{s'}$. Note that in the latter case, $s' = t$. As we assumed $a \in ||\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V$, we find there must be a $b \in \Delta^{\mathcal{G}}$ such that $(a,b) \in r^{\mathcal{G}}$ and $b \in ||\nu X_{\bar{s'}}.\psi''||^{\mathcal{G}}_V$, both if $\psi'' = \psi'$ and if $\psi'' = \varphi$. Thus, $s'(b) \in U$, which means $s'(b) \notin \neg.U$ and moreover, we find $s'(b) \notin S_{j'}$ (I.H. of the previous proposition), thus $b \notin \lceil s' \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}$. As $\lceil \forall r.s' \rceil^{\mathcal{G}}_{S_{j'}} = \{a' \in \Delta^{\mathcal{G}} \mid \forall a' \in \Delta^{\mathcal{G}} : (a,a') \in r^{\mathcal{G}} \text{ implies } a' \in \lceil s' \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}\}$, we must conclude that, because of the witness b, $a \notin \lceil \mathcal{C}(s) \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}$.

- $\mathcal{C}(s) = s' \vee s''$. In this case, the subformula looks like $\nu X_{\bar{s}}.\psi' \wedge \psi''$, for ψ' and ψ'' either a variable Y or a subformula of the form $\nu Y.\chi$. There are two possible options: either $\{s', s''\} \subseteq W$, in which case a similar argument as before suffices. However, if, without loss of generality, $s' \notin X$, the subformula looks like $\nu X_{\bar{s}}.(\nu X_{\bar{s'}}.\psi') \wedge \psi''$ we are back in the case of $\nu X_{\bar{s'}}.\psi \in sub(\varphi)$ such that $X_t \notin sub(\psi)$, and thus $||\nu X_{\bar{s'}}.\psi||^{\mathcal{G}}_V \subseteq \lfloor \neg s' \rfloor^{\mathcal{G}}_{S_{j'}}$, as concluded before. As we assumed $a \in ||\nu X_{\bar{s}}.\psi||^{\mathcal{G}}_V$, this means that also $a \in ||\nu X_{\bar{s'}}.\psi||^{\mathcal{G}}_V$, and thus $a \in \lfloor \neg s' \rfloor^{\mathcal{G}}_{S_{j'}}$. This corresponds to $\neg s'(a) \in S_{j'}$, so

clearly also $a \notin \lceil s' \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}$. To argue why $a \notin \lceil s'' \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}$, a similar argument as above applies, as $s'' \in W$. Thus, we find $a \notin \lceil \mathcal{C}(s) \rceil^{\mathcal{G}}_{S_{j'} \cup \neg.U}$.

- Other cases are treated similarly.

□

Theorem 3. *For each shapes pair $(\mathcal{C}, s)$, each data graph $\mathcal{G}$, and each node c of $\mathcal{G}$, we have*

$$\mathcal{G} \textit{ validates } (\mathcal{C}, s(c)) \quad \textit{iff} \quad \mathcal{G}, c \models tr^{+}_{\emptyset,\mathcal{C}}(s).$$

Proof. First, note that for each $\mathcal{G}$ there exists an $i \in \mathbb{N}$ such that the least fixed point coincides with its approximation: $||\mu X.\varphi||^{\mathcal{G}}_V = ||\mu^i X.\varphi||^{\mathcal{G}}_V$. Similarly, there exists an $i' \in \mathbb{N}$ such that $W^{\uparrow i'}_{\mathcal{G},\mathcal{C}}(\emptyset) = W^{\uparrow j}_{\mathcal{G},\mathcal{C}}(\emptyset)$ for all $j > i'$. Propositions 6 and 7 apply specifically to these i and i', which suffices to conclude the above. □

C Proofs for Section 6 (Complexity Results)

Proposition 4. *For each shapes pair $(\mathcal{C}, s)$ and each guess $\mathbf{G}$ for $\mathcal{C}$, we can define a 2ATA $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ such that:*

1. *If $tr^{+}_{\emptyset,\mathcal{C}}(s)$ is satisfiable, then for some guess $\mathbf{G}$, the language recognised by $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ is non-empty.*
2. *If for some guess $\mathbf{G}$ the language of $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ is not empty, then $tr^{+}_{\emptyset,\mathcal{C}}(s)$ is satisfiable.*
3. *The number of states in $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ is linear in $|sub(\mathcal{C})|$.*

Proof. We use $a_1, \ldots, a_l$ to denote the nominals occurring in $\mathcal{C}$, Furthermore, we assume all input trees are k-ary full trees; all non-leaf nodes have k children, where k is the maximum of the amount of $\langle r \rangle \psi \in sub(\mathcal{C})$, and l plus one.

Let $\mathbf{A}(\mathcal{C}, s, \mathbf{G}) := \langle \Sigma, Q, \delta, \tilde{q}, p \rangle$, be defined as follows. As the automaton construction remains so close to the automaton constructed by Sattler and Vardi (2001), we will just provide the technical details and refer the reader to their work for more intuition and explanation. Let

$$\Sigma = \{\bot, root\} \cup \{\sigma \mid \sigma \subseteq N_C(\mathcal{C}) \cup N_I(\mathcal{C}) \cup \overline{N}_R(\mathcal{C}) \cup \{\xrightarrow{r} a_i \mid r \in \overline{N}_R(\mathcal{C}), 1 \leq i \leq l\}\},$$

where $N_C(\mathcal{C})$, $N_I(\mathcal{C})$ and $\overline{N}_R(\mathcal{C})$ are the restrictions of the alphabets N_C, N_I and $\overline{N}_R$ to the symbols used in $\mathcal{C}$. Let

$$Q = \{\bot, \tilde{q}, q_0, q_0', q_1, \ldots, q_l, q, q', q''\} \cup \{tr^{+}(\varphi), tr^{-}(\varphi) \mid \varphi \in sub(\mathcal{C}) \cup \{\neg s \mid s \in sub(\mathcal{C})\}\} \cup \{r, \neg r \mid r \in \overline{N}_R(\mathcal{C})\}$$

$$p(q) = \begin{cases} 1 & \text{if } q = tr^{+}(\varphi), \varphi \in sub(\mathcal{C}) \\ 0 & \text{otherwise.} \end{cases}$$

Note it is immediate that the number of states in linear in $|sub(\mathcal{C})|$. Finally, we define the transition function in the following way, for each $\sigma \in \Sigma$.

$$\delta(\tilde{q}, \sigma) = (0, q_0) \wedge (0, q_0')$$

$$\delta(q_0, \sigma) = \begin{cases} \bigwedge_{i=1}^{l}(i, q_i) \wedge \bigwedge_{i=l+1}^{k}(i, q) \wedge \bigwedge_{i=1}^{k}(i, q'') & \text{if } root = \sigma \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(q'', \sigma) = \begin{cases} \top & \text{if } r \notin \sigma, r^{-} \notin \sigma, \text{ for each } r \in \overline{N}_R(\mathcal{C}) \\ \bot & \text{otherwise} \end{cases}$$

In the following, let $1 \leq i \leq l$, $\hat{q} \in Q \setminus \{q_0, q_1, \ldots, q_l, q'', q\}$, $r \in \overline{N}_R(\mathcal{C})$, $p \in N_I \cup NC$, $\hat{\sigma} \in \Sigma$, $\sigma \in \Sigma \setminus \{\bot\}$ and moreover

$$\Gamma(i) = \begin{cases} (i, \bot) & \text{if } \gamma_i = \bot \\ \bigwedge_{\varphi \in \gamma_i}(i, tr^{+}(\varphi)) & \text{if } \gamma_i \subseteq sub(\mathcal{C}) \cup \{\neg s \mid s \in sub(\mathcal{C})\} \end{cases}$$

then

$$\delta(q_i,\hat{\sigma}) = \begin{cases} \bigwedge_{i=1}^{k}(i,q) & \text{if } \gamma_i \cap (N_I \cup N_C) = \hat{\sigma} \cap (N_I \cup N_C), root \neq \hat{\sigma}, \\ & \text{and, for each } a \in N_I \cap \hat{\sigma} \text{ and } (a,r,a') \in C, \xrightarrow{r} a' \in \hat{\sigma} \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(q,\hat{\sigma}) = \begin{cases} \bigwedge_{i=1}^{k}(i,q) & \text{if } \hat{\sigma} \cap N_I = \emptyset \text{ and } root \neq \hat{\sigma} \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(\hat{q},\bot) = \begin{cases} \top & \text{if } \hat{q} = \bot \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(\bot,\hat{\sigma}) = \begin{cases} \top & \text{if } \hat{\sigma} = \bot \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(q'_0,\sigma) = \bigwedge_{i=1}^{l}\Gamma(i) \wedge \bigvee_{i=1}^{k}(i,tr^+(s)) \wedge \wedge \bigwedge_{i=1}^{k}((1,q') \vee (i,\bot))$$

$$\delta(q',\sigma) = \bigwedge_{\xrightarrow{r^-}a_i \in \sigma, \forall r.s \in \gamma_{f(i)}} (0,tr^+(s)) \wedge \bigwedge_{\xrightarrow{r^-}a_i \in \sigma, \{\exists r.s, \neg s'\} \subseteq \gamma_{f(i)}, s' \leftarrow \exists r.s \in \mathcal{C}} (0,tr^-(s)) \wedge \bigwedge_{i=1}^{k}((i,q') \vee (i,\bot))$$

$$\delta(r,\sigma) = \begin{cases} \top & \text{if } r \in \sigma \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(\neg r,\sigma) = \begin{cases} \top & \text{if } r \notin \sigma \text{ and } \sigma \neq root \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(tr^+(p),\sigma) = \begin{cases} \top & \text{if } p \in \sigma \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(tr^+(s \wedge s'),\sigma) = (0,tr^+(s)) \wedge (0,tr^+(s'))$$

$$\delta(tr^+(s \vee s'),\sigma) = (0,tr^+(s)) \vee (0,tr^+(s'))$$

$$\delta(tr^+(\neg s),\sigma) = (0,tr^-(s))$$

$$\delta(tr^+(s),\sigma) = (0,tr^+(\mathcal{C}(s)))$$

$$\delta(tr^+(\exists r.s),\sigma) = \begin{cases} \top & \text{if } \xrightarrow{r} a_i \in \sigma, s \in \gamma_{f(i)} \\ \bigvee_{j=1}^{k}((j,tr^+(s)) \wedge (j,r)) & \text{otherwise} \end{cases}$$

$$\delta(tr^+(\forall r.s),\sigma) = \begin{cases} \bot & \text{if } \xrightarrow{r} a_i \in \sigma, s \notin \gamma_{f(i)} \\ ((-1,tr^+(s)) \vee (0,\neg r^-)) \wedge \bigwedge_{j=1}^{k}((j,tr^+(s)) \vee (j,\neg r) \vee (j,\bot)) & \text{otherwise} \end{cases}$$

$$\delta(tr^-(p),\sigma) = \begin{cases} \top & \text{if } p \notin \sigma \text{ and } \sigma \neq root \\ \bot & \text{otherwise} \end{cases}$$

$$\delta(tr^-(s \wedge s'),\sigma) = (0,tr^-(s)) \vee (0,tr^-(s'))$$

$$\delta(tr^-(s \vee s'),\sigma) = (0,tr^-(s)) \wedge (0,tr^-(s'))$$

$$\delta(tr^-(\neg s),\sigma) = (0,tr^+(s))$$

$$\delta(tr^-(s),\sigma) = (0,tr^-(\mathcal{C}(s)))$$

$$\delta(tr^-(\exists r.s),\sigma) = \begin{cases} \bot & \text{if } \xrightarrow{r} a_i \in \sigma, \neg s \notin \gamma_{f(i)} \\ ((-1,tr^-(s)) \vee (0,\neg r^-)) \wedge \bigwedge_{j=1}^{k}((j,tr^-(s)) \vee (j,\neg r) \vee (j,\bot)) & \text{otherwise} \end{cases}$$

$$\delta(tr^-(\forall r.s),\sigma) = \begin{cases} \top & \text{if } \xrightarrow{r} a_i \in \sigma, \neg s \in \gamma_{f(i)} \\ \bigvee_{j=1}^{k}((j,tr^-(s)) \wedge (j,r)) & \text{otherwise.} \end{cases}$$

□

Theorem 7. *Deciding document satisfiability for $\mathcal{ALCIO}$ SHACL is* EXPTIME*-complete, under the well-founded semantics. Finite document satisfiability for $\mathcal{ALCIO}$ SHACL is decidable.*

Proof. The main idea is to take the 2ATA $\mathbf{A}(\mathcal{C}, s, \mathbf{G})$ for some $(c, s) \in \mathcal{T}$, and update the transition function δ in the following way. For every $(D, s) \in \mathcal{T}$, with $D \in N_C \cup \overline{N}_R$, if $D \in \sigma$, the transition function for the state that is traversing the whole tree already, q', is updated with an extra conjunct: $(0, tr^+(s))$. For the individuals $a_i \in N_I$, this can directly be taken care of by adding $(i, tr^+(s))$ as a conjunct to $\delta(q'_0, \sigma)$, where q'_0 is the state that already ensures all nominals in the graph are assigned the right states.

More specifically, taking the exact same automaton as in the proof of Proposition 4, with the following being redefined as described below, suffices.

$$\Gamma(i) = \begin{cases} (i, \bot) & \text{if } \gamma_i = \bot \\ \bigwedge_{\varphi \in \gamma_i} (i, tr^+(\varphi)) \wedge \bigwedge_{(a_i, s) \in \mathcal{T}} (i, tr^+(\varphi)) & \text{if } \gamma_i \subseteq sub(\mathcal{C}) \cup \{\neg s \mid s \in sub(\mathcal{C})\} \end{cases}$$

$$\delta(q', \sigma) = \bigwedge_{\xrightarrow{r^-} a_i \in \sigma, \forall r.s \in \gamma_{f(i)}} (0, tr^+(s)) \wedge \bigwedge_{(D,s) \in \mathcal{T}, D \in (N_C \cup \overline{N}_R) \cap \sigma} (0, tr^+(s))$$

$$\wedge \bigwedge_{\xrightarrow{r^-} a_i \in \sigma, \{\exists r.s, \neg s'\} \subseteq \gamma_{f(i)}, s' \leftarrow \exists r.s \in \mathcal{C}} (0, tr^-(s)) \wedge \bigwedge_{i=1}^{k} ((i, q') \vee (i, \bot))$$

For finite satisfiability, we note decidability follows from among others the combination of Proposition 1, Theorem 4 combined with reasoning analogous to the proof of Theorem 6. □