Expressiveness Results for an Inductive Logic of Separated Relations

Radu Iosif
Univ. Grenoble Alpes, CNRS, Grenoble INP, VERIMAG, 38000, France

Florian Zuleger
Institute of Logic and Computation, Technische Universität Wien, Austria

Abstract

In this paper we study a Separation Logic of Relations (SLR) and compare its expressiveness to (Monadic) Second Order Logic [(M)SO]. SLR is based on the well-known Symbolic Heap fragment of Separation Logic, whose formulæ are composed of points-to assertions, inductively defined predicates, with the separating conjunction as the only logical connective. SLR generalizes the Symbolic Heap fragment by supporting general relational atoms, instead of only points-to assertions. In this paper, we restrict ourselves to finite relational structures, and hence only consider Weak (M)SO, where quantification ranges over finite sets. Our main results are that SLR and MSO are incomparable on structures of unbounded treewidth, while SLR can be embedded in SO in general. Furthermore, MSO becomes a strict subset of SLR, when the treewidth of the models is bounded by a parameter and all vertices attached to some hyperedge belong to the interpretation of a fixed unary relation symbol. We also discuss the problem of identifying a fragment of SLR that is equivalent to MSO over models of bounded treewidth.

2012 ACM Subject Classification Theory of computation → Separation logic

Keywords and phrases Separation Logic, Model Theory, Monadic Second Order Logic, Treewidth

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2023.20

Related Version *Full Version*: https://arxiv.org/abs/2307.02381 [42]

Funding *Radu Iosif*: The first author wishes to acknowledge the support of the French National Research Agency project Non-Aggregative Resource COmpositions (NARCO) under grant number ANR-21-CE48-0011.

1 Introduction

Relational structures are interpretations of relation symbols that define the standard semantics of first and second order logic [58]. They provide a unifying framework for reasoning about a multitude of graph types e.g., graphs with multiple edges, labeled graphs, colored graphs, hypergraphs, etc. Graphs are, in turn, important for many areas of computing, e.g., static analysis [45], databases and knowledge representation [1] and concurrency [27].

A well-established language for specifying graph properties is Monadic Second Order Logic (MSO), where quantification is over vertices only, or both vertices and edges, and sets thereof [25]. Other graph description logics use formal language theory (e.g., regular expressions, context-free grammars) to check for paths with certain patterns [37].

Another way of describing graphs is by an algebra of operations, such as vertex/hyperedge replacement, i.e., substitution of a vertex/hyperedge in a graph by another graph. Graph algebras come with robust notions of *recognizable sets* (i.e., unions of equivalence classes of a finite index congruence) and *inductive sets* (i.e., least solutions of recursive sets of equations, sometimes also called *equational* or *context-free* sets [25]). The relation between the expressivity of MSO-definable, recognizable and inductive sets is well-understood: all definable sets are recognizable, but there are recognizable sets that are not definable [22].

The equivalence between definability and recognizability has been established for those sets in which the *treewidth* (a positive integer that indicates how close the graph is to a tree) is bounded by a fixed constant [6]. Moreover, it is known that the set of graphs of treewidth bounded by a constant is inductive [25, Theorem 2.83].

From a system designer's point of view, logical specification is declarative (i.e., it describes required properties, such as acyclicity, hamiltonicity, etc.), whereas algebraic specification is operational (i.e., describes the way graphs are built from pieces), relying on low-level details (e.g., designated source vertices). Because of this, system provers (e.g., model checkers or deductive verifiers) tend to use logic both for requirement specification and internal representation of configuration sets. However, algebraic theories (e.g., automata theory) are used to obtain algorithms for discharging the generated logical verification conditions, e.g., satisfiability of formulæ or validity of entailments between formulæ.

Separation Logic (SL) [43, 56, 18] is a first order substructural logic with a *separating conjunction* $*$ that decomposes structures. For reasons related to its applications in the deductive verification of pointer-manipulating programs, the models of SL are finite graphs of fixed outdegree, described by partial functions, called *heaps*. The separating conjunction is interpreted in SL as the union of heaps with disjoint domains.

Since their early days, substructural logics have had (abstract) algebraic semantics [54], yet their relation with graph algebras has received scant attention. However, as we argue in this paper, the standard interpretation of the separating conjunction has the flavor of certain graph-algebraic operations, such as the disjoint union with fusion of designated nodes [23].

The benefits of SL over purely boolean graph logics (e.g., MSO) are two-fold:

I. The separating conjunction in combination with inductive definitions [2] provide concise descriptions of datastructures in the heap memory of a program. For instance, the rules (1) $\mathsf{ls}(x, y) \leftarrow x = y$ and (2) $\mathsf{ls}(x, y) \leftarrow \exists z \;.\; x \mapsto z * \mathsf{ls}(z, y)$ define finite singly-linked list segments, that are either (1) empty with equal endpoints, or (2) consist of a single cell x *separated from* the rest of the list segment $\mathsf{ls}(z, y)$. Most recursive datastructures (singly- and doubly-linked lists, trees, etc.) can be defined using only existentially quantified spatial conjunctions of atoms, that are (dis-)equalities and points-to atoms. This simple subset of SL is referred to as the *Symbolic Heap* fragment. The problems of model checking [13], satisfiability [12], robustness properties [44] and entailment [21, 47, 34, 35, 53] for this fragment have been studied extensively.

II. The separating conjunction is a powerful tool for reasoning about mutations of heaps. In fact, the built-in separating conjunction allows to describe actions *locally*, i.e., only with respect to the resources (e.g., memory cells, network nodes) involved, while framing out the part of the state that is irrelevant for that particular action. This principle of describing mutations, known as *local reasoning* [16], is at the heart of very powerful compositional proof techniques for pointer programs using SL [14].

The extension of SL from heaps to relational structures, called *Separation Logic of Relations* (SLR), has been first considered for relational databases and type systems of object-oriented languages, known as role logic [48]. Our motivation for studying the expressivity of SLR arose from several works:

(1) deductive verification of self-adapting distributed systems, where Hoare-style local reasoning is applied to write correctness proofs for systems with dynamically changing network architectures [4, 7, 9], and

(2) model-checking such systems for absence of deadlocks and critical section violations [10].

Another possible application of SLR is reasoning about programs with overlaid datastructures [31, 46], using variants of SL with a per-field composition of heaps, naturally expressed in SLR.

Table 1 A comparison of SLR, MSO and SO in terms of expressiveness, where ✓ means that the inclusion holds, × means it does not and ? denotes an open problem.

	SLR		MSO		SO	
SLR	✓	?	× (§4)	× (§4)	✓(§5)	✓(§7)
MSO	× (§4)	✓(§6)	✓	✓(§7)	✓	✓(§7)
SO	× (§4)	?	× (§7)	× (§7)	✓	✓(§7)

The SLR separating conjunction is understood as splitting the interpretation of each relation symbol from the signature into disjoint parts. For instance, the formula $\mathsf{R}(x_1, \ldots, x_n)$ describes a structure in which all relations are empty and R consists of a single tuple of values $x_1, \ldots, x_n$, whereas $\mathsf{R}(x_1, \ldots, x_n) * \mathsf{R}(y_1, \ldots, y_n)$ says that R consists of two distinct tuples, i.e., the values of x_i and y_i differ for at least one index $1 \leq i \leq n$. In contrast to the Courcelle-style composition of disjoint structures with fusion of nodes that interpret the common constants (i.e., function symbols of arity zero) [23], the SLR-style composition (i.e., the pointwise disjoint union of the interpretations of each relation symbol) is more fine-grained. For instance, if structures are used to encode graphs, SLR allows to specify (hyper-)edges that have no connected vertices, isolated vertices, or both. The same style of composition is found in other spatial logics for graphs, such as the GL logic of Cardelli, Gardner and Ghelli [18].

In particular, SLR is strictly more expressive than standard SL interpreted over heaps. For instance, the previous definition of a list segment can be written in a relational signature having at least a unary relation $\mathfrak{D}$ and a binary relation $\mathfrak{H}$, as (1) $\mathsf{rls}(x, y) \leftarrow x = y$ and (2) $\mathsf{rls}(x, y) \leftarrow \exists z \,.\, \mathfrak{D}(x) * \mathfrak{H}(x, z) * \mathsf{rls}(z, y)$. Note that the $\mathfrak{D}(x)$ atoms joined by separating conjunction ensure that all the nodes are pairwise different, except for the last one denoted by y. We will later generalize this use of $\mathfrak{D}$ for the definition of a Courcelle-style composition operator [23], where $\mathfrak{D}$ ensures that all but a bounded number of nodes are pairwise different. Further, SLR can describe graphs of unbounded degree, e.g., stars with a central vertex and outgoing binary edges E to frontier vertices e.g., (1) $\mathsf{star}(x) \leftarrow \mathfrak{N}(x) * \mathsf{node}(x)$ (2) $\mathsf{node}(x) \leftarrow x = x$ and (3) $\mathsf{node}(x) \leftarrow \exists y \,.\, \mathfrak{E}(x, y) * \mathfrak{N}(y) * \mathsf{node}(x)$. The definition of stars is not possible with SL interpreted over heaps, because of their bounded out-degree.

Our contributions. We compare the expressiveness of SLR with (monadic) second-order logic (M)SO. We are interested in finite relational structures, and hence only consider *weak* (M)SO, where relations are interpreted as finite sets.

For a logic $\mathcal{L} \in \{\mathsf{SLR}, \mathsf{MSO}, \mathsf{SO}\}$ using a finite set Σ of relation and constant symbols, we denote by $[\![\mathcal{L}]\!]$ the set of sets of models for all formulæ $\phi \in \mathcal{L}$. For a unary relation symbol $\mathfrak{D}$ not in Σ, considered fixed in the rest of the paper, we say that a graph is *guarded* if all elements from a tuple in the interpretation of a relation symbol belong to the interpretation of $\mathfrak{D}$. Then $[\![\mathcal{L}]\!]^{\mathfrak{D},k}$ is the set of sets of guarded models of treewidth at most k of a formula from $\mathcal{L}$, where the signature of $\mathcal{L}$ is extended with $\mathfrak{D}$, and $[\![\mathcal{L}_1]\!]^{\mathfrak{D},k} \subseteq [\![\mathcal{L}_2]\!]$ means that $\mathcal{L}_2$ is at least as expressive as $\mathcal{L}_1$, when only guarded models of treewidth at most k are considered. Note that $[\![\mathcal{L}]\!]^{\mathfrak{D},k} \subseteq [\![\mathcal{L}]\!]$ is not a trivial statement, in general, because it asserts the existence of a formula of $\mathcal{L}$ that defines the set of guarded structures of treewidth at most k.

Each cell of Table 1 shows $[\![\mathcal{L}_1]\!] \subseteq [\![\mathcal{L}_2]\!]$ (left) and $[\![\mathcal{L}_1]\!]^{\mathfrak{D},k} \subseteq [\![\mathcal{L}_2]\!]$ (right). Here ✓ means that the inclusion holds, × means it does not and ? denotes an open problem, with reference to the sections where (non-trivial) proofs are given. The most interesting cases are:

1. SLR and MSO are incomparable on unguarded structures of unbounded treewidth, i.e., there are formulæ in each of the logics that do not have an equivalent in the other,

2. SO is strictly more expressive than SLR, when considering unguarded structures of unbounded treewidth, and at least as expressive as SLR, when considering guarded structures of bounded treewidth,
3. SLR is strictly more expressive than MSO, when considering guarded structures of bounded treewidth; this shows the expressive power of SLR, emphasizing (once more) the model-theoretic importance of the treewidth parameter.

Note that, when considering SLR-definable sets of bounded treewidth, we systematically assume these structures to be guarded. We state as an open problem and conjecture that every infinite SLR-definable set of structures of bounded treewidth is necessarily guarded, in a hope that the guardedness condition can actually be lifted. So far, similar conditions have been used to, e.g., obtain decidability of entailments between SL symbolic heaps [41, 47] and of invariance for assertions written in a fragment of SLR for verifying distributed networks [9]. Moreover, the problem of checking if a given set of inductive definitions defines a guarded set of structures is decidable for these logics [44, 8].

A further natural question asks for a fragment of SLR with the same expressive power as MSO, over structures of bounded treewidth. This is also motivated by the need for a general fragment of SLR with a decidable entailment problem, that is instrumental in designing automated verification systems. Unfortunately, such a definition is challenging because the MSO-definability of the sets defined by SLR is an undecidable problem, whereas treewidth boundedness of such sets remains an open problem, conjectured to be decidable.

All proofs can be found in the full version of the paper [42].

Related work. Treewidth is a cornerstone of algorithmic tractability. For instance, many NP-complete graph problems such as Hamiltonicity and 3-Coloring become PTIME, when restricted to inputs whose treewidth is bounded by a constant, see, e.g., [38, Chapter 11]. Moreover, bounding the treewidth by a constant sets the frontier between the decidability and undecidability of monadic second order (MSO) logical theories. A result of Courcelle [22] proves that MSO is decidable over bounded treewidth structures, by reduction to the emptiness problem of tree automata. A dual result of Seese [57] proves that each class of structures with a decidable MSO theory necessarily has bounded treewidth.

Comparing the expressiveness of SL [56] with classical logics received a fair amount of attention. A first proof of undecidability of the satisfiability problem for SL, with first order quantification, negation and separating implication, but without inductive definitions [17], is based on a reduction to Trakhtenbrot's undecidability result for first order logic on finite models [32]. This proof uses heaps of outdegree two to encode arbitrary binary relations as $\mathsf{R}(x, y) \stackrel{\text{def}}{=} \exists z \,.\, z \mapsto (x, y) * \mathsf{true}$. A more refined proof for heaps of outdegree one was given in [11], where it was shown that SO has the same expressivity as SL, when negation and separating implication is allowed, which is not the case for our fragment of SLR.

A related line of work, pioneered by Lozes [50], is the translation of quantifier-free SL formulæ into boolean combinations of *core formulæ*, belonging to a small set of very simple patterns. This enables a straightforward translation of the quantifier-free fragment of SL into first order logic, over unrestricted signatures with both relation and function symbols, subsequently extended to two quantified variables [28] and restricted quantifier prefixes [33]. Moreover, a translation of quantifier-free SL into first order logic, based on the small model property of the former, has been described in [15]. These are fragments of SL without inductive definitions, but with arbitrary combinations of boolean (conjunction, negation) and spatial (separating conjunction, magic wand) connectives. A non-trivial attempt of generalizing the technique of core formulæ to *reachability* and *list segment* predicates is given in [29]. Moreover, an in-depth comparison between the expressiveness of various models of separation, i.e., spatial, as in SL, and contextual (subtree-like), as in Ambient Logic [19], can be found in [52]. The restriction of SLR on trees is, however, out of the scope of this paper.

An early combination of spatial connective for graph decomposition with (least fixpoint) recursion is Graph Logic (GL) [18], whose expressiveness is compared to that of MSO_2, i.e., MSO interpreted over graphs, with quantification over both vertices and edges [26]. For reasons related to its applications, GL quantifies over the vertices and edge labels of a graph, unlike MSO_2 that quantifies over vertices, edges and sets thereof. Another fairly subtle difference is that GL can describe graphs with multiple edges that involve the same vertices and same label, whereas the models of MSO_2 are simple graphs. Without recursion, GL can be translated into MSO_2 and it has been shown that MSO_2 is strictly more expressive than GL without edge label quantification [5]. Little is known for GL with recursion, besides that it can express PSPACE-complete model checking problems [26], whereas model checking is PSPACE-complete for MSO [59].

The separating conjunction used in SLR has been first introduced in role logic [48], a logic designed to reason about properties of record fields in object-oriented programs. This logic uses separating conjunction in combination with boolean connectives and first order quantifier (ranging over vertices) and has no recursive constructs (least fixpoints or inductive definitions). A bothways translation between role logic and SO has been described in [49]. These translations rely on boolean connectives and first order quantifiers, instead of least fixpoint recursion, which is the case in our work.

To complete the picture, a substructural logic with separating conjunction and implication, based on a layered decomposition of graphs has been developped in [20]. However, the relation between this logic and (M)SO remains unexplored, to the best of our knowledge.

2 Definitions

For a set A, we denote by $\mathrm{pow}(A)$ its powerset, $A^1 \stackrel{\text{def}}{=} A$, $A^{i+1} \stackrel{\text{def}}{=} A^i \times A$, for all $i \geq 1$, where $\times$ is the Cartesian product, and $A^+ \stackrel{\text{def}}{=} \bigcup_{i \geq 1} A^i$. The cardinality of a finite set A is denoted by $\|A\|$. Given integers i and j, we write $[i, j]$ for the set $\{i, i+1, \ldots, j\}$, empty if $i > j$. For a partial function $f : A \to B$, we denote by $\mathrm{dom}(f)$ its domain and by $f\downharpoonright_S$ its restriction to $S \subseteq \mathrm{dom}(f)$. f is *locally co-finite* iff the set $\{a \in A \mid f(a) = b\}$ is finite, for all $b \in B$. f is *effectively computable* iff there exists a Turing machine $\mathcal{M}$, such that, for any $a \in \mathrm{dom}(f)$, $\mathcal{M}$ outputs $f(a)$ in finitely many steps and diverges for $a \notin \mathrm{dom}(f)$.

Signatures and Structures. Let $\Sigma = \{\mathsf{R}_1, \ldots, \mathsf{R}_N, \mathsf{c}_1, \ldots, \mathsf{c}_M\}$ be a finite *signature*, where R_i are relation symbols of arity $\#\mathsf{R}_i \geq 1$ and c_j are constant symbols, i.e., function symbols of arity zero. Additionally, we assume the existence of a unary relation symbol $\mathfrak{D}$, not in Σ. Unless stated otherwise, we consider Σ and $\mathfrak{D}$ to be fixed in the following.

A *structure* is a pair (U, σ), where U is an *infinite* set, called *universe*, and $\sigma : \Sigma \to U \cup \mathrm{pow}(U^+)$ is an *interpretation* that maps each relation symbol R to a relation $\sigma(\mathsf{R}) \subseteq U^{\#\mathsf{R}}$ and each constant c to an element $\sigma(\mathsf{c}) \in U$. Two structures are *isomorphic* iff they differ only by a renaming of their elements (a formal definition is given in, e.g., [32, §A3]). We write $\mathrm{Rel}(\sigma)$ for the set of elements that belong to $\sigma(\mathsf{R})$, for some relation symbol $\mathsf{R} \in \Sigma$ and $\mathrm{Supp}(\sigma) \stackrel{\text{def}}{=} \mathrm{Rel}(\sigma) \cup \{\sigma(\mathsf{c}_1), \ldots, \sigma(\mathsf{c}_M)\}$ for the *support* of the structure, that includes the interpretation of constants. We denote by $\mathit{Str}(\Sigma)$ (resp. $\mathit{Str}(\Sigma, \mathfrak{D})$) the set of structures over the signature Σ (resp. $\Sigma \cup \{\mathfrak{D}\}$).

A structure is *guarded* iff all nodes that occur in some tuple from the denotation of a relation symbol sit also inside the denotation of the unary relation $\mathfrak{D}$:

▶ **Definition 1.** *A structure* $(U, \sigma) \in Str(\Sigma, \mathfrak{D})$ *is* guarded *iff* $\mathrm{Rel}(\sigma) = \sigma(\mathfrak{D})$.

Two interpretations σ_1 and σ_2 are *compatible* iff $\sigma_1(\mathsf{c}) = \sigma_2(\mathsf{c})$, for all constant symbols $\mathsf{c} \in \Sigma$. Two structures (U_1, σ_1) and (U_2, σ_2) are *locally disjoint* iff $\sigma_1(\mathsf{R}) \cap \sigma_2(\mathsf{R}) = \emptyset$, for all relation symbols $\mathsf{R} \in \Sigma$. The (spatial) *composition* of structures is defined below:

▶ **Definition 2.** *The* composition *of two compatible and locally disjoint structures* (U_1, σ_1) *and* (U_2, σ_2) *is* $(U_1, \sigma_1) \bullet (U_2, \sigma_2) \stackrel{\text{def}}{=} (U_1 \cup U_2, \sigma_1 \uplus \sigma_2)$*, where* $(\sigma_1 \uplus \sigma_2)(\mathsf{R}_i) \stackrel{\text{def}}{=} \sigma_1(\mathsf{R}_i) \cup \sigma_2(\mathsf{R}_i)$ *and* $(\sigma_1 \uplus \sigma_2)(\mathsf{c}_j) \stackrel{\text{def}}{=} \sigma_1(\mathsf{c}_j) = \sigma_2(\mathsf{c}_j)$*, for all* $i \in [1, N]$ *and* $j \in [1, M]$*. The composition is undefined for structures that are not compatible or not locally disjoint.*

Graphs and Treewidth. A graph is a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, such that $\mathcal{V}$ is a set of *vertices* and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of *edges*. All graphs considered in this paper are finite and directed, i.e., $\mathcal{E}$ is not necessarily a symmetric relation. Graphs are naturally encoded as structures:

▶ **Definition 3.** *A graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *is encoded by the structure* $(U_\mathcal{G}, \sigma_\mathcal{G})$ *over the signature* $\Gamma \stackrel{\text{def}}{=} \{\mathfrak{V}, \mathfrak{E}\}$*, where* $\#\mathfrak{V} = 1$ *and* $\#\mathfrak{E} = 2$*, such that* $U_\mathcal{G} = \mathcal{V}$*,* $\sigma_\mathcal{G}(\mathfrak{V}) = \mathcal{V}$ *and* $\sigma_\mathcal{G}(\mathfrak{E}) = \mathcal{E}$*.*

A *path* in $\mathcal{G}$ is a sequence of pairwise distinct vertices $v_1, \ldots, v_n$, such that $(v_i, v_{i+1}) \in \mathcal{E}$, for all $i \in [1, n-1]$. We say that $v_1, \ldots, v_n$ is an *undirected path* if $\{(v_i, v_{i+1}), (v_{i+1}, v_i)\} \cap \mathcal{E} \neq \emptyset$ instead, for all $i \in [1, n-1]$. A set of vertices $V \subseteq \mathcal{V}$ is *connected in* $\mathcal{G}$ iff there is an undirected path in $\mathcal{G}$ between any two vertices in V. A graph $\mathcal{G}$ is *connected* iff $\mathcal{V}$ is connected in $\mathcal{G}$. A *clique* is a graph such that each two distinct nodes are the endpoints of an edge, the direction of which is not important. We denote by $\mathcal{K}_n$ the set of cliques with n vertices.

Given a set Λ of labels, a Λ-*labeled tree* is a tuple $\mathcal{T} = (\mathcal{N}, \mathcal{F}, r, \lambda)$, where $(\mathcal{N}, \mathcal{F})$ is a graph, $r \in \mathcal{N}$ is a designated vertex called the *root*, such that there exists a unique path in $(\mathcal{N}, \mathcal{F})$ from r to any other vertex $v \in \mathcal{N} \setminus \{r\}$ and r has no incoming edges $(p, r) \in \mathcal{F}$. The mapping $\lambda : \mathcal{N} \to \Lambda$ associates each vertex of the tree a *label* from Λ.

▶ **Definition 4.** *A* tree decomposition *of a structure* (U, σ) *over the signature* Σ *is a* $\mathrm{pow}(U)$*-labeled tree* $\mathcal{T} = (\mathcal{N}, \mathcal{F}, r, \lambda)$*, such that the following hold:*

1. *for each relation symbol* $\mathsf{R} \in \Sigma$ *and each tuple* $\langle u_1, \ldots, u_{\#\mathsf{R}} \rangle \in \sigma(\mathsf{R})$ *there exists* $n \in \mathcal{N}$*, such that* $\{u_1, \ldots, u_{\#\mathsf{R}}\} \subseteq \lambda(n)$*, and*
2. *for each* $u \in \mathrm{Supp}(\sigma)$*, the set* $\{n \in \mathcal{N} \mid u \in \lambda(n)\}$ *is nonempty and connected in* $(\mathcal{N}, \mathcal{F})$*.*

The width *of the tree decomposition is* $\mathrm{tw}(\mathcal{T}) \stackrel{\text{def}}{=} \max_{n \in \mathcal{N}} \|\lambda(n)\| - 1$*. The* treewidth *of the structure* (U, σ) *is* $\mathrm{tw}(U, \sigma) \stackrel{\text{def}}{=} \min\{\mathrm{tw}(\mathcal{T}) \mid \mathcal{T}$ *is a tree decomposition of* $\sigma\}$*.*

A set of structures is *treewidth-bounded* iff the set of corresponding treewidths is finite and *treewidth-unbounded* otherwise. A set is *strictly treewidth-unbounded* iff it is treewidth-unbounded and any of its infinite subsets is treewidth-unbounded. The following result can be found in [30, Theorem 12.3.9] and is restated here for self-containment:

▶ **Proposition 5.** *The set of cliques* $\{\mathcal{K}_n \mid n \in \mathbb{N}\}$ *is strictly treewidth-unbounded.*

3 Logics

We introduce two logics over a relational signature $\Sigma = \{\mathsf{R}_1, \ldots, \mathsf{R}_N, \mathsf{c}_1, \ldots, \mathsf{c}_M\}$. First, the *Separation Logic of Relations* (SLR) uses a set of *first order variables* $\mathbb{V}_1 = \{x, \ldots\}$ and a set of *predicates* $\mathbb{A} = \{\mathsf{A}, \ldots\}$ (also called *recursion variables* in the literature, e.g., [18]) of arities $\#\mathsf{A} \geq 0$. We use the symbols $\xi, \chi \in \mathbb{V}_1 \cup \{\mathsf{c}_1, \ldots, \mathsf{c}_M\}$ to denote *terms*, i.e., either first order variables or constants. The formulæ of SLR are defined by the following syntax:

$$\phi := \mathsf{emp} \mid \xi = \chi \mid \xi \neq \chi \mid \mathsf{R}(\xi_1, \ldots, \xi_{\#\mathsf{R}}) \mid \mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}}) \mid \phi * \phi \mid \exists x \,.\, \phi$$

The formulæ $\xi = \chi$ and $\xi \neq \chi$ are called *equalities* and *disequalities*, $\mathsf{R}(\xi_1, \ldots, \xi_{\#\mathsf{R}})$ and $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$ are called *relation* and *predicate atoms*, respectively. A formula with no occurrences of predicate atoms (resp. existential quantifiers) is called *predicate-free* (resp. *quantifier-free*). A variable is *free* if it does not occur within the scope of an existential quantifier and *bound* otherwise. We denote by $\mathrm{fv}(\phi)$ be the set of free variables of ϕ. A *sentence* is a formula with no free variables. A *substitution* $\phi[x_1/\xi_1 \ldots x_n/\xi_n]$ replaces simultaneously every occurrence of the free variable x_i by the term ξ_i in ϕ, for all $i \in [1, n]$. As a convention, the bound variables in ϕ are renamed to avoid clashes with $\xi_1, \ldots, \xi_n$.

The predicates from $\mathbb{A}$ are interpreted as sets of structures, defined inductively:

▶ **Definition 6.** *A* set of inductive definitions (SID) Δ *is a* finite *set of* rules *of the form* $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi$*, where* $x_1, \ldots, x_{\#\mathsf{A}}$ *are pairwise distinct variables, called* parameters*, such that* $\mathrm{fv}(\phi) \subseteq \{x_1, \ldots, x_{\#\mathsf{A}}\}$*. A rule* $\mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi$ *is said to* define A*.*

The semantics of SLR formulæ is given by the satisfaction relation $(U, \sigma) \models^\nu_\Delta \phi$ between structures and formulæ. This relation is parameterized by a *store* $\nu : \mathbb{V}_1 \to U$ mapping the free variables of a formula into elements of the universe and an SID Δ. We write $\nu[x \leftarrow u]$ for the store that maps x into u and agrees with ν on all variables other than x. For a term ξ, we denote by $(\sigma, \nu)(\xi)$ the value $\sigma(\xi)$ if ξ is a constant, or $\nu(\xi)$ if ξ is a first-order variable. The satisfaction relation is the least relation that satisfies the following conditions:

$$\begin{array}{lcl}
(U,\sigma) \models^\nu_\Delta \mathsf{emp} & \Leftrightarrow & \sigma(\mathsf{R}) = \emptyset, \text{ for all } \mathsf{R} \in \Sigma \\
(U,\sigma) \models^\nu_\Delta \xi \sim \chi & \Leftrightarrow & (U,\sigma) \models^\nu_\Delta \mathsf{emp} \text{ and } (\sigma,\nu)(\xi) \sim (\sigma,\nu)(\chi), \text{ where } \sim\, \in \{=, \neq\} \\
(U,\sigma) \models^\nu_\Delta \mathsf{R}(\xi_1, \ldots, \xi_{\#\mathsf{R}}) & \Leftrightarrow & \sigma(\mathsf{R}) = \{\langle (\sigma,\nu)(\xi_1), \ldots, (\sigma,\nu)(\xi_{\#\mathsf{R}}) \rangle\} \\
& & \quad \text{and } \sigma(\mathsf{R}') = \emptyset, \text{for } \mathsf{R}' \in \Sigma \setminus \{\mathsf{R}\} \\
(U,\sigma) \models^\nu_\Delta \mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}}) & \Leftrightarrow & (U,\sigma) \models^\nu_\Delta \phi[x_1/\xi_1, \ldots, x_{\#\mathsf{A}}/\xi_{\#\mathsf{A}}], \\
& & \quad \text{for some } \mathsf{A}(x_1, \ldots, x_{\#\mathsf{A}}) \leftarrow \phi \in \Delta \\
(U,\sigma) \models^\nu_\Delta \phi_1 * \phi_2 & \Leftrightarrow & \text{there exist structures } (U_1,\sigma_1) \text{ and } (U_2,\sigma_2), \text{ such that} \\
& & (U,\sigma) = (U_1,\sigma_1) \bullet (U_2,\sigma_2) \text{ and } (U,\sigma_i) \models^\nu_\Delta \phi_i, \text{ for } i = 1,2 \\
(U,\sigma) \models^\nu_\Delta \exists x \,.\, \phi & \Leftrightarrow & (U,\sigma) \models^{\nu[x \leftarrow u]}_\Delta \phi, \text{ for some } u \in U
\end{array}$$

Note that every structure (U, σ), such that $(U, \sigma) \models^\nu_\Delta \phi$, interprets each relation symbol as a finite set of tuples, defined by a finite least fixpoint iteration over the rules in Δ. In particular, the assumption that each universe is infinite excludes the cases in which a SLR formula becomes unsatisfiable because the universe does not have enough elements to be assigned to the existentially quantified variables during the unfolding of the rules.

If ϕ is a sentence, the satisfaction relation does not depend on the store, in which case we write $(U, \sigma) \models_\Delta \phi$ and say that (U, σ) is a Δ-model of ϕ. We denote by $[\![\phi]\!]_\Delta$ the set of Δ-models of ϕ. We call $[\![\phi]\!]_\Delta$ an SLR-*definable* set. By $[\![\phi]\!]^{\mathfrak{D},k}_\Delta$ we denote the set of guarded structures (Def. 1) of treewidth at most k from $[\![\phi]\!]_\Delta$. We write $[\![\mathsf{SLR}]\!] \stackrel{\text{def}}{=} \{[\![\phi]\!]_\Delta \mid \phi \text{ is a SLR formula}, \Delta \text{ is a SID}\}$ and $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \stackrel{\text{def}}{=} \{[\![\phi]\!]^{\mathfrak{D},k}_\Delta \mid \phi \text{ is a SLR formula}, \Delta \text{ is a SID}\}$. Below we show that SLR-definable sets are unions of isomorphic equivalence classes:

▶ **Proposition 7.** *Given isomorphic structures (U, σ) and (U', σ'), for any sentence ϕ of SLR and any SID Δ, we have $(U, \sigma) \models_\Delta \phi \Leftrightarrow (U', \sigma') \models_\Delta \phi$.*

The other logic is the *Weak Second Order Logic* (SO) defined using a set of *second order variables* $\mathbb{V}_2 = \{X, \ldots\}$, in addition to first order variables $\mathbb{V}_1$. We denote by $\#X$ the arity of a second order variable X. Terms and atoms are the same as in SLR. The formulæ of SO have the following syntax:

$$\psi := \xi = \chi \mid \mathsf{R}(\xi_1, \ldots, \xi_{\#\mathsf{R}}) \mid X(\xi_1, \ldots, \xi_{\#X}) \mid \neg\psi \mid \psi \wedge \psi \mid \exists x \,.\, \psi \mid \exists X \,.\, \psi$$

We write $\xi \neq \chi \stackrel{\text{def}}{=} \neg\xi = \chi$, $\psi_1 \vee \psi_2 \stackrel{\text{def}}{=} \neg(\neg\psi_1 \wedge \neg\psi_2)$, $\psi_1 \rightarrow \psi_2 \stackrel{\text{def}}{=} \neg\psi_1 \vee \psi_2$, $\forall x \,.\, \psi \stackrel{\text{def}}{=} \neg\exists x \,.\, \neg\psi$ and $\forall X \,.\, \psi \stackrel{\text{def}}{=} \neg\exists X \,.\, \neg\psi$. The Weak Monadic Second Order Logic (MSO) is the fragment of SO restricted to second-order variables of arity one. The Weak Existential Second Order Logic (ESO) is the fragment of SO consisting of formulæ of the form $\exists X_1 \ldots \exists X_n \,.\, \phi$, where ϕ has only first order quantifiers.

The semantics of SO is given by a relation $(U, \sigma) \Vdash^\nu \psi$, where the store $\nu : \mathbb{V}_1 \cup \mathbb{V}_2 \rightarrow U \cup \mathrm{pow}(U^+)$ maps each first-order variable $x \in \mathbb{V}_1$ to an element of the universe $\nu(x) \in U$ and each second-order variable $X \in \mathbb{V}_2$ to a finite relation $\nu(X) \subseteq U^{\#X}$. The satisfaction relation of SO is defined inductively on the structure of formulæ:

$$\begin{array}{lcl}
(U, \sigma) \Vdash^\nu \xi = \chi & \Leftrightarrow & (\sigma, \nu)(\xi) = (\sigma, \nu)(\chi) \\
(U, \sigma) \Vdash^\nu \mathsf{R}(\xi_1, \ldots, \xi_{\#\mathsf{R}}) & \Leftrightarrow & \langle (\sigma, \nu)(\xi_1), \ldots, (\sigma, \nu)(\xi_{\#\mathsf{R}}) \rangle \in \sigma(\mathsf{R}) \\
(U, \sigma) \Vdash^\nu X(\xi_1, \ldots, \xi_{\#X}) & \Leftrightarrow & \langle (\sigma, \nu)(\xi_1), \ldots, (\sigma, \nu)(\xi_{\#X}) \rangle \in \nu(X) \\
(U, \sigma) \Vdash^\nu \exists X \,.\, \psi & \Leftrightarrow & (U, \sigma) \Vdash^{\nu[X \leftarrow V]} \psi, \text{ for some finite set } V \subseteq U^{\#X}
\end{array}$$

The semantics of negation, conjunction and first-order quantification are standard and omitted for brevity. Note the difference between equalities and relation atoms in SLR and SO: in the former, equalities (relation atoms) hold in an empty (singleton) structure, whereas no such upper bounds on the cardinality of the model of an atom occur in SO.

However, SO can express upper bounds on the cardinality of the universe. Such formulæ are unsatisfiable under the assumption that the universe of each structure is infinite. We chose to keep the comparison between SLR and SO simple and not consider the general case of a finite universe, for the time being. A detailed study of SL interpreted over finite universe heaps, with arbitrary nesting of boolean and separating connectives but without inductive definitions is given in [33]. We plan to give a similar comparison in an extended version.

If ϕ is a sentence, we write $(U, \sigma) \Vdash \phi$ instead of $(U, \sigma) \Vdash^\nu \phi$ and define $[\![\phi]\!] \stackrel{\text{def}}{=} \{(U, \sigma) \mid (U, \sigma) \Vdash \phi\}$ and $[\![\phi]\!]^{\mathfrak{D},k}$ for the restriction of $[\![\phi]\!]$ to guarded structures of treewidth at most k. We call $[\![\phi]\!]$ an (M)SO-*definable* set. We write $[\![\text{(M)SO}]\!] \stackrel{\text{def}}{=} \{[\![\phi]\!] \mid \phi \text{ is a (M)SO formula}\}$ and $[\![\text{(M)SO}]\!]^{\mathfrak{D},k} \stackrel{\text{def}}{=} \{[\![\phi]\!]^{\mathfrak{D},k} \mid \phi \text{ is a (M)SO formula}\}$.

The aim of this paper is comparing the expressive powers of SLR, MSO and SO, with respect to the properties that can be defined in these logics. We are concerned with the problems $[\![\mathcal{L}_1]\!] \subseteq [\![\mathcal{L}_2]\!]$ and $[\![\mathcal{L}_1]\!]^{\mathfrak{D},k} \subseteq [\![\mathcal{L}_2]\!]$, where $\mathcal{L}_1$ and $\mathcal{L}_2$ are any of the logics SLR, MSO and SO, respectively. In particular, for $[\![\mathcal{L}_1]\!]^{\mathfrak{D},k} \subseteq [\![\mathcal{L}_2]\!]$, we implicitly assume that $\mathcal{L}_1$ and $\mathcal{L}_2$ are sets of formulæ over the relational signature $\Sigma \cup \{\mathfrak{D}\}$. Table 1 summarizes our results, with references to the sections in the paper where the (non-trivial) proofs can be found, and the remaining open problems.

4 $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!] \not\subseteq [\![\mathsf{SLR}]\!]$

The argument that shows $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!]$ is that MSO cannot express the fact that the cardinality of a set is even [22, Proposition 6.2]. The SLR rules below state that the cardinality of $\mathfrak{R}$ is even, for a predicate A of arity zero:

$$\mathsf{A}() \leftarrow \exists x \exists y \,.\, \mathfrak{R}(x) * \mathfrak{R}(y) * A() \qquad \mathsf{A}() \leftarrow \mathsf{emp}$$

Note that every model of $\mathsf{A}()$ interprets $\mathfrak{R}$ as a set with an even number of disconnected elements and every other relation symbol by an empty set. The treewidth of such models is one, thus $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!]$ for any $k \geq 1$, and we obtain $[\![\mathsf{SLR}]\!] \not\subseteq [\![\mathsf{MSO}]\!]$, in general.

The argument for $[\![\mathsf{MSO}]\!] \not\subseteq [\![\mathsf{SLR}]\!]$ is that the set of cliques is MSO-definable (actually, even first order definable) but not SLR-definable. First, the set $\{\mathcal{K}_n \mid n \in \mathbb{N}\}$ is defined by the following first order formula in the signature of graph encodings (Def. 3):

$$\forall x \forall y \,.\, \mathfrak{V}(x) \wedge \mathfrak{V}(y) \wedge x \neq y \rightarrow \mathfrak{E}(x,y) \vee \mathfrak{E}(y,x)$$

Since this set is strictly treewidth-unbounded (Prop. 5), it is sufficient to prove that SLR cannot define strictly treewidth-unbounded sets. More precisely, for each SLR sentence ϕ and SID Δ, we prove the existence of an integer $W \geq 1$, depending on ϕ and Δ alone, such that

(i) for each structure $(U,\sigma) \in [\![\phi]\!]_\Delta$ there exists a structure $(U,\overline{\sigma}) \in [\![\phi]\!]_\Delta$, of treewidth at most W, and

(ii) the function that maps (U,σ) into $(U,\overline{\sigma})$ is locally co-finite (Lemma 10).

Then each infinite SLR-definable set has an infinite treewidth-bounded subset, i.e., it is not strictly treewidth-unbounded (Prop. 12).

A first ingredient of the proof is that each SID can be transformed into an equivalent SID without equality constraints between variables:

▶ **Definition 8.** *A rule $\mathsf{A}(x_1,\ldots,x_{\#\mathsf{A}}) \leftarrow \exists y_1 \ldots \exists y_n \,.\, \psi$, where ψ is a quantifier-free formula, is* normalized *iff no equality atom $x = y$ occurs in ψ, for distinct variables $x, y \in \{x_1,\ldots,x_{\#\mathsf{A}}\} \cup \{y_1,\ldots,y_n\}$. An SID is* normalized *iff it contains only normalized rules.*

▶ **Lemma 9.** *Given an SID Δ, one can build a normalized SID Δ' such that, for each structure σ and each predicate atom $\mathsf{A}(\xi_1,\ldots,\xi_{\#\mathsf{A}})$, we have $(U,\sigma) \models_\Delta \exists \xi_{i_1} \ldots \exists \xi_{i_n} \,.\, \mathsf{A}(\xi_1,\ldots,\xi_{\#\mathsf{A}}) \Leftrightarrow (U,\sigma) \models_{\Delta'} \exists \xi_{i_1} \ldots \exists \xi_{i_n} \,.\, \mathsf{A}(\xi_1,\ldots,\xi_{\#\mathsf{A}})$, where $\{\xi_{i_1},\ldots,\xi_{i_n}\} = \{\xi_1,\ldots,\xi_{\#\mathsf{A}}\} \cap \mathbb{V}_1$.*

A consequence is that, in the absence of equality constraints, each existentially quantified variable instantiated by the inductive definition of the satisfaction relation can be assigned a distinct element of the universe. For instance, considering the rules $\mathsf{fold\_ls}(x_1) \leftarrow \mathsf{emp}$ and $\mathsf{fold\_ls}(x_1) \leftarrow \exists y \,.\, \mathfrak{H}(x_1,y) * \mathsf{fold\_ls}(y)$, the $\mathsf{fold\_ls}(x)$ formula defines an infinite set of graphs whose edges are given by the interpretation of a relation symbol $\mathfrak{H}$, such that there exists an Eulerian path visiting all edges exactly once, and all vertices possibly more than once. Since there are no equality constraints, each model of $\mathsf{fold\_ls}(x)$ can be expanded into an acyclic list that never visits the same vertex twice, except at the endpoints. This graph has treewidth two, if the endpoints coincide, and one otherwise.

Formally, we write $(U,\sigma) \overset{\bullet}{\models}{}^\nu_\Delta \phi$ iff the satisfaction relation $(U,\sigma) \models^\nu_\Delta \phi$ can be established by considering finite injective stores. The definition of $\overset{\bullet}{\models}{}^\nu_\Delta$ is the same as the one of $\models^\nu_\Delta$ (§3), except for the cases below:

$$\begin{aligned}
(U,\sigma) \overset{\bullet}{\models}{}^\nu_\Delta \phi_1 * \phi_2 \;&\Leftrightarrow \text{there exist structures } (U_1,\sigma_1) \bullet (U_2,\sigma_2) = (U,\sigma)\text{, such that} \\
&\qquad U_1 \cap U_2 = \nu(\mathrm{fv}(\phi_1) \cap \mathrm{fv}(\phi_2)) \text{ and } (U_i,\sigma_i) \overset{\bullet}{\models}{}^{\nu|_{\mathrm{fv}(\phi_i)}}_\Delta \phi_i\text{, for } i = 1,2 \\
(U,\sigma) \overset{\bullet}{\models}{}^\nu_\Delta \exists x \,.\, \phi \;&\Leftrightarrow (U,\sigma) \overset{\bullet}{\models}{}^{\nu[x \leftarrow u]}_\Delta \phi\text{, for some } u \in U \setminus \nu(\mathrm{fv}(\phi))
\end{aligned}$$

For instance, we have $(U,\sigma) \overset{\bullet}{\models}{}^\nu_\Delta \mathsf{fold\_ls}(x)$ only if $\sigma(\mathfrak{H})$ is a list of pairwise distinct elements.

▶ **Lemma 10.** *Given a normalized SID Δ, a predicate atom $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$, for each structure (U, σ) and a store ν, such that $(U, \sigma) \models^{\nu}_{\Delta} \mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$, there exists a structure $(U, \overline{\sigma})$, such that $(U, \overline{\sigma}) \overset{\bullet}{\models}{}^{\nu}_{\Delta} \mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$. Moreover, the function with domain $[\![\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})]\!]_{\Delta}$ that maps (U, σ) into the set of structures isomorphic with $(U, \overline{\sigma})$ is locally co-finite.*

We show that the models defined on injective stores have bounded treewidth:

▶ **Lemma 11.** *Given a normalized SID Δ and a predicate atom $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$, we have $\mathrm{tw}(\sigma) \leq W$, for each structure (U, σ) and store ν, such that $(U, \sigma) \overset{\bullet}{\models}{}^{\nu}_{\Delta} \mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$, where $W \geq 1$ is a constant depending only on Δ.*

Note that proving Lemmas 10 and 11 for predicate atoms loses no generality, because for each formula ϕ, such that $\mathrm{fv}(\phi) = \{x_1, \ldots, x_n\}$, we can consider a predicate symbol A_ϕ of arity n and extend the SID by the rule $\mathsf{A}_\phi(x_1, \ldots, x_n) \leftarrow \phi$. The proof of $[\![\mathsf{MSO}]\!] \not\subseteq [\![\mathsf{SLR}]\!]$ relies on the following:

▶ **Proposition 12.** *Given a sentence ϕ and an SID Δ, $[\![\phi]\!]_{\Delta}$ is either finite or it has an infinite subset of bounded treewidth.*

5 $[\![\mathsf{SLR}]\!] \subseteq [\![\mathsf{SO}]\!]$

Since SLR and MSO are incomparable, it is natural to ask for a logic that subsumes both of them. In this section, we prove that SO is such a logic. Since MSO is a syntactic subset of SO, we have $[\![\mathsf{MSO}]\!] \subseteq [\![\mathsf{SO}]\!]$ trivially. We show that $[\![\mathsf{SLR}]\!] \subseteq [\![\mathsf{SO}]\!]$ using the fact that each model of a predicate atom in SLR is built according to a *finite unfolding tree* indicating the partial order in which the rules of the SID are used in the inductive definition of the satisfaction relation; in other words, unfolding trees are for SIDs what derivation trees are for context-free grammars. More precisely, any model of a SLR sentence can be decomposed into pairwise disjoint substructures, each being the model of the quantifier- and predicate-free subformula of a rule in the SID, such that there is a one-to-one mapping between the nodes of the tree and the substructures from the decomposition of the model. We use second order variables, interpreted as finite relations, to define the unfolding tree and the mapping between the nodes of the unfolding tree and the tuples in the interpretation of the relation symbols from the model. These second order variables are existentially quantified and the resulting SO formula describes the model, without the unfolding tree that witnesses its construction according to the rules of the SID.

Let $\Delta \overset{\text{def}}{=} \{\mathsf{r}_1, \ldots, \mathsf{r}_R\}$ be a given SID. Without loss of generality, for each relation symbol $\mathsf{R} \in \Sigma$, we assume that there is at most one occurrence of an atom $\mathsf{R}(y_1, \ldots, y_{\#\mathsf{R}})$ in each rule from Δ. If this is not the case, we split the rule by introducing a new predicate symbol for each relation atom with relation symbol R_i, until the condition is satisfied.

▶ **Definition 13.** *An* unfolding tree *for a predicate atom $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$ is a Δ-labeled tree $\mathcal{T} = (\mathcal{N}, \mathcal{F}, r, \lambda)$, such that $\lambda(r)$ defines A and, for each vertex $n \in \mathcal{N}$, if $\mathsf{B}_1(z_{1,1}, \ldots, z_{1,\#\mathsf{B}_1})$, $\ldots$, $\mathsf{B}_h(z_{h,1}, \ldots, z_{h,\#\mathsf{B}_h})$ are the predicate atoms that occur in $\lambda(n)$, then $p_1, \ldots, p_h$ are the children of n in $\mathcal{T}$, such that $\lambda(p_\ell)$ defines B_ℓ, for all $\ell \in [1, h]$.*

We build a SO formula that defines the models of a relation atom $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$. As explained above, this is without loss of generality. Let P be the maximum number of occurrences of predicate atoms in a rule from Δ_ϕ. We use second order variables $Y_1, \ldots, Y_P$ of arity 2, for the edges of the unfolding tree and $X_1, \ldots, X_R$ of arity 1, for the labels of the nodes in the unfolding tree, i.e., the rules of Δ. First, we build a SO formula $\mathfrak{T}(x, \{X_i\}_{i=1}^{R}, \{Y_j\}_{j=1}^{P})$, as the conjunction of SO formulæ that describe the following facts:

- the root x belongs to X_i, for some rule r_i that defines A,
- the sets $X_1, \ldots, X_R$ are pairwise disjoint,
- each vertex in $X_1 \cup \ldots \cup X_R$ is reachable from x by a path with edges $Y_1, \ldots, Y_P$,
- each vertex in $X_1 \cup \ldots \cup X_R$, except for x, has exactly one incoming edge,
- x has no incoming edge,
- each vertex from X_i has exactly h outgoing edges $Y_1, \ldots, Y_h$, each to a vertex from X_{j_ℓ}, respectively, such that r_{j_ℓ} defines B_ℓ, for all $\ell \in [1,h]$, where $\mathsf{B}_1(z_{1,1}, \ldots, z_{1,\#\mathsf{B}_1}), \ldots, \mathsf{B}_h(z_{h,1}, \ldots, z_{h,\#\mathsf{B}_h})$ are the predicate atoms that occur in r_i.

Second, we build a SO formula expressing the relationship between the unfolding tree $\mathcal{T} = (\mathcal{N}, \mathcal{F}, r, \lambda)$ and the model. The formula $\mathfrak{F}(\xi_1, \ldots, \xi_{\#\mathsf{A}}, x, \{X_i\}_{i=1}^R, \{Y_j\}_{j=1}^P, \{\{Z_{k,\ell}\}_{\ell=1}^{\#\mathsf{R}_k}\}_{k=1}^N)$ uses second order variables $Z_{k,\ell}$, of arity 2, that encode partial functions mapping a tree node n to the value of ξ_ℓ for the (unique) atom $\mathsf{R}_k(\xi_1, \ldots, \xi_{\#\mathsf{R}_i})$ from the rule $\lambda(n)$, in case such an atom exists. The formula $\mathfrak{F}$ is the conjunction of following SO-definable facts[1]:

(i) each second order variable $Z_{k,\ell}$ denotes a functional binary relation,

(ii) for each tree node labeled by a rule r_i and each atom $\mathsf{R}_k(\xi_1, \ldots, \xi_{\#\mathsf{R}_k})$ occurring at that node, the interpretation of R_k contains a tuple, whose elements are related to the node via $Z_{k,1}, \ldots, Z_{k,\#\mathsf{R}_k}$, respectively,

(iii) for any (not necessarily distinct) rules r_i and r_j such that an atom with relation symbol R_k occurs in both, the corresponding tuples from the interpretation of R_k are distinct,

(iv) each tuple from the interpretation of R_k must have been introduced by a relation atom with relation symbol R_k that occurs in a rule r_i,

(v) two terms ξ_m and χ_n that occur in two relation atoms $\mathsf{R}_k(\xi_1, \ldots, \xi_{\#\mathsf{R}_k})$ and $\mathsf{R}_\ell(\chi_1, \ldots, \chi_{\#\mathsf{R}_\ell})$ within rules r_i and r_j, respectively, and are constrained to be equal (i.e., via equalities and parameter passing), must be equated,

(vi) a disequality $\xi \neq \chi$ that occurs in a rule r_i is propagated throughout the tree to each pair of variables that occur within two relation atoms $\mathsf{R}_k(\xi_1, \ldots, \xi_{\#\mathsf{R}_k})$ and $\mathsf{R}_\ell(\chi_1, \ldots, \chi_{\#\mathsf{R}_\ell})$ in rules r_{j_k} and r_{j_ℓ}, respectively, such that ξ is bound ξ_r and χ to χ_s by equalities and parameter passing,

(vii) each term in $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$ that is bound to a variable from a relation atom $\mathsf{R}_k(z_1, \ldots, z_{\#\mathsf{R}_k})$ in the unfolding, must be equated to that variable.

Summing up, the SO formula defining the models of the predicate atom $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$ with respect to the SID Δ is:

$$\begin{aligned}&\mathfrak{A}^{\mathsf{A}}_\Delta(\xi_1, \ldots, \xi_{\#\mathsf{A}}) \stackrel{\mathsf{def}}{=} \exists x \exists \{X_i\}_{i=1}^R \exists \{Y_j\}_{j=1}^P \exists \{Z_{1,\ell}\}_{\ell=1}^{\#\mathsf{R}_1} \ldots \exists \{Z_{K,\ell}\}_{\ell=1}^{\#\mathsf{R}_K} \,. \\ &\mathfrak{T}(x, \{X_i\}_{i=1}^R, \{Y_j\}_{j=1}^P) \wedge\ \mathfrak{F}(\xi_1, \ldots, \xi_{\#\mathsf{A}}, x, \{X_i\}_{i=1}^R, \{Y_j\}_{j=1}^P, \{\{Z_{k,\ell}\}_{\ell=1}^{\#\mathsf{R}_k}\}_{k=1}^N)\end{aligned}$$

The correctness of the above construction is proved in the following proposition, that also shows $[\![\mathsf{SLR}]\!] \subseteq [\![\mathsf{SO}]\!]$:

▶ **Proposition 14.** *Given an SID Δ and a predicate atom $\mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}})$, for each structure (U, σ) and store ν, we have $(U, \sigma) \models^\nu_\Delta \mathsf{A}(\xi_1, \ldots, \xi_{\#\mathsf{A}}) \Leftrightarrow (U, \sigma) \Vdash^\nu \mathfrak{A}^{\mathsf{A}}_\Delta(\xi_1, \ldots, \xi_{\#\mathsf{A}})$.*

We state as an open question whether the above formula can be written in ESO, which would sharpen the comparison between SLR and SO, as ESO is known to be strictly less expressive than SO [40]. In particular, the problem is writing $\mathfrak{F}$ in ESO.

[1] The exact SO formulæ are given in the full version of the paper [42].

6 $[\![\mathsf{MSO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$

We prove that, for any MSO sentence ϕ and any integer $k \geq 1$, there exists an SID $\Delta(k,\phi)$ and a predicate $\mathsf{A}_{k,\phi}$ of arity zero, such that $[\![\phi]\!]^{\mathfrak{D},k} = [\![\mathsf{A}_{k,\phi}()]\!]_{\Delta(k,\phi)}$, i.e., the set of guarded models of ϕ of treewidth at most k corresponds to the set of structures SLR-defined by the predicate atom $\mathsf{A}_{k,\phi}()$, when interpreted in the SID $\Delta(k,\phi)$. Our proof leverages from a technique of Courcelle [23], used to show that the models of bounded treewidth of a given MSO sentence can be described by a finite set of recursive equations, written using an algebra of operations on structures. This result follows up in a long-standing line of work (known as Feferman-Vaught theorems [51]) that reduces the evaluation of an MSO sentence on the result of an algebraic operation to the evaluation of several related sentences in the arguments of the respective operation.

6.1 A Theorem of Courcelle

We recall first a result of Courcelle [23], that describes the structures of bounded treewidth, which satisfy a given MSO formula ϕ, by an effectively constructible set of recursive equations. This set of equations uses two operations on structures, namely *glue* and fgcst_j, that are lifted to sets of structures, as usual. The result is developed in two steps. The first step builds a generic set of equations, that characterizes all structures of treewidth at most k. This set of equations is then refined, in the second step, to describe only models of ϕ. Because this result applies to general (i.e., finite and infinite) structures (U,σ), we do not require U to be infinite, for the purposes of this presentation. We consider a fixed integer $k \geq 1$ and MSO sentence ϕ in the rest of this section.

Operations on Structures. Let Σ_1 and Σ_2 be two (possibly overlapping) signatures. The *glueing* operation $\mathit{glue} : \mathit{Str}(\Sigma_1) \times \mathit{Str}(\Sigma_2) \to \mathit{Str}(\Sigma_1 \cup \Sigma_2)$ is the union of structures with *disjoint universes*, followed by fusion of the elements denoted by constants. Formally, given $S_i = (U_i, \sigma_i)$, for $i = 1,2$, such that $U_1 \cap U_2 = \emptyset$, let $\sim$ be the least equivalence relation on $U_1 \cup U_2$ such that $\sigma_1(\mathsf{c}) \sim \sigma_2(\mathsf{c})$, for all $\mathsf{c} \in \Sigma_1 \cap \Sigma_2$. Let $[u]$ be the equivalence class of $u \in U_1 \cup U_2$ with respect to $\sim$ and lift this notation to tuples and sets of tuples. Then $\mathit{glue}(S_1,S_2) \stackrel{\text{def}}{=} (U,\sigma)$, where $U \stackrel{\text{def}}{=} \{[u] \mid u \in U_1 \cup U_2\}$ and σ is defined as follows:

$$\sigma(\mathsf{R}) \stackrel{\text{def}}{=} \begin{cases} [\sigma_i(\mathsf{R})], & \text{if } \mathsf{R} \in \Sigma_i \setminus \Sigma_{3-i}, \text{ for both } i = 1,2 \\ [\sigma_1(\mathsf{R}) \cup \sigma_2(\mathsf{R})], & \text{if } \mathsf{R} \in \Sigma_1 \cap \Sigma_2 \end{cases}$$

Since we match isomorphic structures, the nature of the elements of U (i.e., equivalence classes) is not important. The *forget* operation $\mathit{fgcst}_j : \mathit{Str}(\Sigma) \to \mathit{Str}(\Sigma \setminus \{\mathsf{c}_j\})$ simply drops the constant c_j from the domain of its argument.

Structures of Bounded Treewidth. Let $\Sigma = \{\mathsf{R}_1, \ldots, \mathsf{R}_N, \mathsf{c}_1, \ldots, \mathsf{c}_M\}$ be a signature and $\Pi = \{\mathsf{c}_{M+1}, \ldots, \mathsf{c}_{M+k+1}\}$ be a set of constants disjoint from Σ, called *ports*. We consider variables Y_i, for all subsets $\Pi_i \subseteq \Pi$, denoting sets of structures over the signature $\Sigma \cup \Pi_i$. The *equation system* $\mathit{Tw}(k)$ is the set of recursive equations of the form $Y_0 \supseteq f(Y_1, \ldots, Y_n)$, where each f is either *glue*, fgcst_{M+j}, for any $j \in [1, k+1]$, or a singleton relation of type R_i, consisting of a tuple with at most $k+1$ distinct elements, for any $i \in [1,N]$. It is known that the set of structures of treewidth at most k is a component of the least solution of $\mathit{Tw}(k)$, in the domain of tuples of sets ordered by pointwise inclusion [25, Theorem 2.83].

Models of MSO Formulæ. The *quantifier rank* $\mathrm{qr}(\phi)$ of an MSO formula ϕ is the maximal depth of nested quantifiers, i.e., $\mathrm{qr}(\phi) \stackrel{\text{def}}{=} 0$ if ϕ is an atom, $\mathrm{qr}(\neg\phi_1) \stackrel{\text{def}}{=} \mathrm{qr}(\phi_1)$, $\mathrm{qr}(\phi_1 \wedge \phi_2) \stackrel{\text{def}}{=} \max(\mathrm{qr}(\phi_1), \mathrm{qr}(\phi_2))$ and $\mathrm{qr}(\exists x \,.\, \phi_1) = \mathrm{qr}(\exists X \,.\, \phi_1) \stackrel{\text{def}}{=} \mathrm{qr}(\phi_1) + 1$. We denote by $\mathbb{F}^r_{\mathsf{MSO}}$ the set of MSO sentences of quantifier rank at most r. This set is finite, up to logical equivalence. For a structure $S = (U, \sigma)$, we define its r-*type* as $\mathit{type}^r(S) \stackrel{\text{def}}{=} \{\phi \in \mathbb{F}^r_{\mathsf{MSO}} \mid S \Vdash \phi\}$. We assume the sentences in $\mathit{type}^r(S)$ to use the signature over which S is defined; this signature will be clear from the context in the following.

▶ **Definition 15.** *An operation $f : \mathit{Str}(\Sigma_1) \times \ldots \times \mathit{Str}(\Sigma_n) \rightarrow \mathit{Str}(\Sigma_{n+1})$ is (effectively) MSO-*compatible[2] *iff, for all structures $S_1, \ldots, S_n$, $\mathit{type}^r(f(S_1, \ldots, S_n))$ depends only on (and can be effectively computed from) $\mathit{type}^r(S_1), \ldots, \mathit{type}^r(S_n)$ by an* abstract operation $f^\sharp : (\mathrm{pow}(\mathbb{F}^r_{\mathsf{MSO}}))^n \rightarrow \mathrm{pow}(\mathbb{F}^r_{\mathsf{MSO}})$.

The result of Courcelle establishes that glueing and forgetting of constants are effectively MSO-compatible operations, with effectively computable abstract operations $\mathit{glue}^\sharp$ and $\mathit{fgcst}^\sharp_{M+i}$, for $i \in [1, k+1]$, see [23, Lemmas 3.2 and 3.3]. As a consequence, one can build from $\mathit{Tw}(k)$ a set of recursive equations $\mathit{Tw}^\sharp(k)$ of the form $Y_0^{\tau_0} = f(Y_1^{\tau_1}, \ldots, Y_n^{\tau_n})$, where $Y_0 = f(Y_1, \ldots, Y_n)$ is an equation from $\mathit{Tw}(k)$ and $\tau_0, \ldots, \tau_n$ are r-types such that $\tau_0 = f^\sharp(\tau_1, \ldots, \tau_n)$. Intuitively, each annotated variable Y^τ denotes the set of structures whose r-type is τ, from the Y-component of the least solution of $\mathit{Tw}(k)$. Given some formula ϕ with $\mathrm{qr}(\phi) = r$, the set of models of ϕ of treewidth at most k is the union of the Y^τ-components of the least solution of $\mathit{Tw}^\sharp(k)$, such that $\phi \in \tau$ [23, Theorem 3.6].

6.2 Encoding Types in SLR

We begin explaining the proof for $[\![\mathsf{MSO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$. Instead of using the set of recursive equations $\mathit{Tw}(k)$ from the previous subsection, we give an SID $\Delta(k)$ that characterizes the guarded structures of bounded treewidth (Fig. 1a). We use the separating conjunction to simulate the glueing operation. The main problem is with the interpretation of the separating conjunction, as composition of structures with possibly overlapping universes (Def. 2), that cannot be glued directly. Our solution is to consider guarded structures (Def. 1), where the unary relation symbol $\mathfrak{D}$ is used to enforce disjointness of the arguments of the composition operation, in all but finitely many elements. Intuitively, $\mathfrak{D}$ "collects" the values assigned to the existentially quantified variables created by rule (2) of $\Delta(k)$ and the top-level rule (4) during the unraveling. This ensures that

(i) the variables of a predicate atom are mapped to pairwise distinct values and
(ii) the composition of two guarded structures is the same as glueing them.

Similar conditions have been used to define e.g., fragments of SL with nice computational properties, such as the *establishment* condition used to ensure decidability of entailments [36], or the *tightness* condition from [4, §5.2].

To alleviate the presentation, the SID $\Delta(k)$ defines only structures $(U, \sigma) \in \mathit{Str}(\Sigma, \mathfrak{D})$ with at least $k+1$ distinct elements in $\sigma(\mathfrak{D})$ (rule 4) and $\sigma(\mathsf{R}) \neq \emptyset$ for at least one relation symbol $\mathsf{R} \in \Sigma$ (rule 3). The cases of structures such that $\|\sigma(\mathfrak{D})\| \leq k$ or $\bigcup_{\mathsf{R} \in \Sigma} \sigma(\mathsf{R}) = \emptyset$ can be dealt with easily, by adding more rules to $\Delta(k)$. In the rest of this section we show that $\Delta(k)$ defines all structures of k-bounded treewidth (except for the mentioned corner cases).

The main property of $\Delta(k)$ is stated below:

▶ **Lemma 16.** *For any guarded structure $(U, \sigma) \in \mathit{Str}(\Sigma, \mathfrak{D})$, such that $\|\sigma(\mathfrak{D})\| \geq k+1$ and $\sigma(\mathsf{R}) \neq \emptyset$, for at least some $\mathsf{R} \in \Sigma$, we have $\mathrm{tw}(\sigma) \leq k$ iff $(U, \sigma) \models_{\Delta(k)} \mathsf{A}_k()$.*

[2] Also referred to as *smooth* operations in [51].

$$\mathsf{A}(x_1,\ldots,x_{k+1}) \leftarrow \mathsf{A}(x_1,\ldots,x_{k+1}) * \mathsf{A}(x_1,\ldots,x_{k+1}) \tag{1}$$

$$\mathsf{A}(x_1,\ldots,x_{k+1}) \leftarrow \exists y \;.\; \mathfrak{D}(y) * \mathsf{A}(x_1,\ldots,x_{k+1})[x_i/y] \text{ for all } i \in [1,k+1] \tag{2}$$

$$\mathsf{A}(x_1,\ldots,x_{k+1}) \leftarrow \mathsf{R}(y_1,\ldots,y_{\#\mathsf{R}}) \text{ for all } \mathsf{R} \in \Sigma \text{ and } y_1,\ldots,y_{\#\mathsf{R}} \in \{x_1,\ldots,x_{k+1}\} \tag{3}$$

$$\mathsf{A}_k() \leftarrow \exists x_1 \ldots \exists x_{k+1} \;.\; \mathfrak{D}(x_1) * \ldots * \mathfrak{D}(x_{k+1}) * \mathsf{A}(x_1,\ldots,x_{k+1}) \tag{4}$$

(a)

$$\mathsf{A}^{\tau}(x_1,\ldots,x_{k+1}) \leftarrow \mathsf{A}^{\tau_1}(x_1,\ldots,x_{k+1}) * \mathsf{A}^{\tau_2}(x_1,\ldots,x_{k+1}) \text{ where } \tau = \mathit{glue}^{\sharp}(\tau_1,\tau_2) \tag{5}$$

$$\begin{aligned}\mathsf{A}^{\tau}(x_1,\ldots,x_{k+1}) \leftarrow & \exists y \;.\; \mathfrak{D}(y) * \mathsf{A}^{\tau_1}(x_1,\ldots,x_{k+1})[x_i/y] \text{ for all } i \in [1,k+1], \text{where} \\ & \tau = \mathit{glue}^{\sharp}(\mathit{fgcst}^{\sharp}_{M+i}(\tau_1),\rho_i) \text{ and } \rho_i \text{ is the type of some structure} \\ & S \in \mathit{Str}(\{\mathsf{c}_{M+i}\},\mathfrak{D}) \text{ with singleton universe and } S \Vdash \mathfrak{D}(\mathsf{c}_{M+i})\end{aligned} \tag{6}$$

$$\begin{aligned}\mathsf{A}^{\tau}(x_1,\ldots,x_{k+1}) \leftarrow & \mathsf{R}(y_1,\ldots,y_{\#\mathsf{R}}) \text{ for some } y_1,\ldots,y_{\#\mathsf{R}} \in \{x_1,\ldots,x_{k+1}\}, \text{ where} \\ & \tau = \mathit{type}^{\mathrm{qr}(\phi)}(S),\; S \in \mathit{Str}(\Sigma \cup \{\mathsf{c}_{M+1},\ldots,\mathsf{c}_{M+k+1}\},\mathfrak{D}) \text{ and} \\ & S \Vdash \mathsf{R}(y_1,\ldots,y_{\#\mathsf{R}})[x_1/\mathsf{c}_{M+1},\ldots,x_{k+1}/\mathsf{c}_{M+k+1}] * \mathop{\Huge *}_{i=1}^{k+1}\mathfrak{D}(\mathsf{c}_{M+i})\end{aligned} \tag{7}$$

$$\begin{aligned}\mathsf{A}_{k,\phi}() \leftarrow & \exists x_1 \ldots \exists x_{k+1} \;.\; \mathfrak{D}(x_1) * \ldots * \mathfrak{D}(x_{k+1}) \;\; * \mathsf{A}^{\tau}(x_1,\ldots,x_{k+1}) \\ & \text{for all } \tau \text{ such that } \phi \in \tau\end{aligned} \tag{8}$$

(b)

Figure 1 The SID $\Delta(k)$ defining structures of treewidth at most k (a) and its annotation $\Delta(k,\phi)$ defining the models of an MSO sentence ϕ, of treewidth at most k (b).

We remark that the encoding of *glue* and fgcst_j used in the definition of $\Delta(k)$ can be used to show that any inductive set of structures, i.e., a set defined by finitely many recursive equations written using *glue* and fgcst_j, can be also defined in SLR. This means that SLR is at least as expressive than the inductive sets, which are always of bounded treewidth.

The second step of our construction is the annotation of the rules in $\Delta(k)$ with $\mathrm{qr}(\phi)$-types, in order to obtain an SID $\Delta(k,\phi)$ (Fig. 1b) describing the models of an MSO sentence ϕ, of treewidth at most k. We consider the set of ports $\Pi = \{\mathsf{c}_{M+1},\ldots,\mathsf{c}_{M+k+1}\}$ disjoint from Σ. The encoding of the store values of the variables $x_1,\ldots,x_{k+1}$ in a given structure is defined below:

▶ **Definition 17.** *Let $\Sigma = \{\mathsf{R}_1,\ldots,\mathsf{R}_N,\mathsf{c}_1,\ldots,\mathsf{c}_M\}$ be a signature, $\Pi = \{\mathsf{c}_{M+1},\ldots,\mathsf{c}_{M+k+1}\}$ be a set of constants not in Σ, and let $(U,\sigma) \in \mathit{Str}(\Sigma,\mathfrak{D})$ be a structure. Let ν be a store mapping $x_1,\ldots,x_{k+1}$ to elements of $U \setminus \sigma(\mathfrak{D})$. Then, $\mathit{encode}((U,\sigma),\nu) \in \mathit{Str}(\Sigma\cup\Pi,\mathfrak{D})$ is a structure with universe U that agrees with (U,σ) over Σ, maps each c_{M+i} to $\nu(x_i)$, for $i \in [1,k+1]$ and maps $\mathfrak{D}$ to $\sigma(\mathfrak{D}) \cup \{\nu(x_1),\ldots,\nu(x_{k+1})\}$.*

The correctness of our construction relies on the fact that the composition acts like glueing, for structures with universe U, whose sets of elements involved in the interpretation of some relation symbol may only overlap at the interpretation of the ports from Π:

▶ **Lemma 18.** *For an integer $r \geq 0$, a store ν and locally disjoint compatible structures $(U_1,\sigma_1),(U_2,\sigma_2) \in \mathit{Str}(\Sigma\cup\Pi,\mathfrak{D})$, such that $\mathrm{Rel}(\sigma_1) \cap \mathrm{Rel}(\sigma_2) \subseteq \{\sigma_1(\mathsf{c}_{M+1}),\ldots,\sigma_1(\mathsf{c}_{M+k+1})\}$ and $(\sigma_1(\mathfrak{D})\cup\sigma_2(\mathfrak{D})) \cap \{\nu(x_i) \mid i \in [1,m]\} = \emptyset$, we have:*

$$\mathit{type}^r(\mathit{encode}((U_1,\sigma_1)\bullet(U_2,\sigma_2),\nu)) = \mathit{glue}^{\sharp}(\mathit{type}^r(\mathit{encode}((U_1,\sigma_1),\nu)),\mathit{type}^r(\mathit{encode}((U_2,\sigma_2),\nu)))$$

Finally, the main property of $\Delta(k,\phi)$ is stated and proved below:

▶ **Proposition 19.** *For any $k \geq 1$, MSO sentence ϕ, and guarded structure $(U,\sigma) \in Str(\Sigma,\mathfrak{D})$, the following are equivalent:*

(1) $(U,\sigma) \Vdash \phi$ *and* $\mathrm{tw}(\sigma) \leq k$, *and*

(2) $(U,\sigma) \models_{\Delta(k,\phi)} \mathsf{A}_{k,\phi}()$.

The above result shows that SLR can define the guarded models $(U,\sigma) \in Str(\Sigma,\mathfrak{D})$ of a given MSO formula whose treewidth is bounded by a given integer. We do not know, for the moment, if this result holds on unguarded structures as well.

The above construction of the SID $\Delta(k,\phi)$ is effectively computable, except for the rule (7), where one needs to determine the type of a structure $S = (U,\sigma)$ with infinite universe. However, we prove in the following that determining this type can be reduced to computing the type of a finite structure, which amounts to solving finitely many MSO model checking problems on finite structures, each of which being PSPACE-complete [59]. Given an integer $n \geq 0$ and a structure $S = (U,\sigma) \in Str(\Sigma)$, we define the finite structure $S^n = (\mathrm{Supp}(\sigma) \cup \{v_1,\ldots,v_n\},\sigma)$, for pairwise distinct elements $v_1,\ldots,v_n \in U \setminus \mathrm{Supp}(\sigma)$. Then, for any quantifier rank r, the structures S and S^{2^r} have the same r-type:

▶ **Lemma 20.** *Given $r \geq 0$ and $S = (U,\sigma) \in Str(\Sigma)$, we have $type^r(S) = type^r(S^{2^r})$.*

As a final remark, we notice that the idea used to prove $[\![\mathsf{MSO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$ can be extended to show also $[\![\mathsf{CMSO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$, where CMSO denotes the extension of MSO with cardinality constraints $\|X\|_{p,q}$ stating that the cardinality of a set of vertices X equals p modulo q, for some constants $0 \leq p < q$. This is because glueing and forgetting constants are CMSO-compatible operations [22, Lemma 4.5, 4.6 and 4.7].

7 The Remaining Cases

We discuss the results from Table 1, that are not already covered by §4, §5 and §6.

$[\![\mathsf{SO}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!]$. Since $[\![\mathsf{SLR}]\!] \subseteq [\![\mathsf{SO}]\!]$ and$[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!]$, we obtain that $[\![\mathsf{SO}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!]$. Moreover, $[\![\mathsf{SO}]\!] \not\subseteq [\![\mathsf{MSO}]\!]$ follows from the fact that our counterexample for $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \not\subseteq [\![\mathsf{MSO}]\!]$ involves only structures of treewidth one.

$[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SO}]\!]$. By applying the translation of SLR to SO from §5 to $\Delta(k)$ (Fig. 1a) and to a given SID Δ defining a predicate A of zero arity, respectively, and taking the conjunction of the results with the SO formula defining guarded structures[3], we obtain an SO formula that defines the set $[\![\mathsf{A}()]\!]_{\Delta}^{\mathfrak{D},k}$, thus proving that $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SO}]\!]$.

$[\![\mathsf{(M)SO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{(M)SO}]\!]$. For each given $k \geq 1$, there exists an MSO formula θ_k that defines the structures of treewidth at most k [25, Proposition 5.11]. This is a consequence of the Graph Minor Theorem proved by Robertson and Seymour [55], combined with the fact that bounded treewidth graphs are closed under taking minors and that the property of having a given finite minor is MSO-definable[4]. Then, for any given (M)SO formula ϕ, the (M)SO formula $\phi \wedge \theta_k$ defines the models of ϕ of treewidth at most k.

[3] $\bigwedge_{\mathsf{R}\in\Sigma} \forall x_1 \ldots \forall x_{\#\mathsf{R}} \,.\, \mathsf{R}(x_1,\ldots,x_{\#\mathsf{R}}) \rightarrow \bigwedge_{i\in[1,\#\mathsf{R}]} \mathfrak{D}(x_i)$.

[4] The proof of Robertson and Seymour does not build θ_k, see [3] for an effective proof.

Open Problems. The following problems from Table 1 are currently open: $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$ and $[\![\mathsf{SO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$, both conjectured to have a negative answer. In particular, the difficulty concerning $[\![\mathsf{SLR}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$ is that, in order to ensure treewidth boundedness, it seems necessary to force the composition of structures to behave like glueing (see the definition of $\Delta(k)$ in Fig. 1a), which seems difficult without the additional relation symbol $\mathfrak{D}$.

Since $[\![\mathsf{MSO}]\!]^{\mathfrak{D},k} \subseteq [\![\mathsf{SLR}]\!]$ but $[\![\mathsf{MSO}]\!] \not\subseteq [\![\mathsf{SLR}]\!]$, we naturally ask for the existence of a fragment of SLR that describes only MSO-definable families of structures of bounded treewidth. In particular, [8, §6] defines a fragment of SLR that has bounded-treewidth models and is MSO-definable. However, in general, since SLR can define context-free sets of guarded graphs (the grammar in Figure 1a can be adapted to encode Hyperedge Replacement (HR) grammars [24]), the MSO-definability of a SLR-definable set is undecidable, as a consequence of the undecidability of the recognizability of context-free languages [39]. On the other hand, the treewidth-boundedness of a SLR-definable set is an open problem, that we conjecture decidable.

A possible direction for future work is also adding Boolean connectives to SLR. Here, one might study an SLR variant that supports Boolean connectives in a top-level logic but not within the inductive definitions, similar to the SL studied in [47, 53]. Adding Boolean connectives within the inductive definitions appears more difficult, as one will need to impose syntactic restitutions such as positive occurrences of predicate atoms in the right hand side of definitions or stratification of negation in order to ensure well-definedness.

8 Conclusions

We have compared the expressiveness of SLR, MSO and SO, in general and for models of bounded treewidth. Interestingly, we found that SLR and MSO are, in general, incomparable and subsumed by SO, whereas the models of bounded treewidth of MSO can be defined by SLR, modulo augmenting the signature with a unary relation symbol used to store the elements that occur in the original structure.

References

1 Serge Abiteboul, Peter Buneman, and Dan Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, 2000.

2 Peter Aczel. An introduction to inductive definitions. In *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, pages 739–782. Elsevier, 1977. doi:10.1016/S0049-237X(08)71120-0.

3 Isolde Adler, Martin Grohe, and Stephan Kreutzer. Computing excluded minors. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 641–650, USA, 2008. Society for Industrial and Applied Mathematics.

4 Emma Ahrens, Marius Bozga, Radu Iosif, and Joost-Pieter Katoen. Reasoning about distributed reconfigurable systems. *Proc. ACM Program. Lang.*, 6(OOPSLA2):145–174, 2022. doi:10.1145/3563293.

5 Timos Antonopoulos and Anuj Dawar. Separating graph logic from mso. In Luca de Alfaro, editor, *Foundations of Software Science and Computational Structures*, pages 63–77, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

6 Mikołaj Bojańczyk and Michał Pilipczuk. Definability equals recognizability for graphs of bounded treewidth. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '16, pages 407–416, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2933575.2934508.

7 Marius Bozga, Lucas Bueri, and Radu Iosif. Decision problems in a logic for reasoning about reconfigurable distributed systems. In Jasmin Blanchette, Laura Kovács, and Dirk Pattinson, editors, *Automated Reasoning – 11th International Joint Conference, IJCAR 2022, Haifa, Israel, August 8-10, 2022, Proceedings*, volume 13385 of *Lecture Notes in Computer Science*, pages 691–711. Springer, 2022. `doi:10.1007/978-3-031-10769-6_40`.

8 Marius Bozga, Lucas Bueri, and Radu Iosif. Decision problems in a logic for reasoning about reconfigurable distributed systems. *CoRR*, abs/2202.09637, 2022. `arXiv:2202.09637`.

9 Marius Bozga, Lucas Bueri, and Radu Iosif. On an invariance problem for parameterized concurrent systems. In Bartek Klin, Slawomir Lasota, and Anca Muscholl, editors, *33rd International Conference on Concurrency Theory, CONCUR 2022, September 12-16, 2022, Warsaw, Poland*, volume 243 of *LIPIcs*, pages 24:1–24:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.CONCUR.2022.24`.

10 Marius Bozga, Radu Iosif, and Joseph Sifakis. Verification of component-based systems with recursive architectures. *Theor. Comput. Sci.*, 940(Part):146–175, 2023. `doi:10.1016/j.tcs.2022.10.022`.

11 Rémi Brochenin, Stéphane Demri, and Étienne Lozes. On the almighty wand. *Inf. Comput.*, 211:106–137, 2012.

12 James Brotherston, Carsten Fuhs, Juan Antonio Navarro Pérez, and Nikos Gorogiannis. A decision procedure for satisfiability in separation logic with inductive predicates. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14–18, 2014*, pages 25:1–25:10. ACM, 2014. `doi:10.1145/2603088.2603091`.

13 James Brotherston, Nikos Gorogiannis, Max I. Kanovich, and Reuben Rowe. Model checking for symbolic-heap separation logic with inductive predicates. In Rastislav Bodík and Rupak Majumdar, editors, *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20–22, 2016*, pages 84–96. ACM, 2016. `doi:10.1145/2837614.2837621`.

14 Cristiano Calcagno, Dino Distefano, Peter W. O'Hearn, and Hongseok Yang. Compositional shape analysis by means of bi-abduction. *J. ACM*, 58(6), December 2011. `doi:10.1145/2049697.2049700`.

15 Cristiano Calcagno, Philippa Gardner, and Matthew Hague. From separation logic to first-order logic. In *Foundations of Software Science and Computational Structures*, pages 395–409, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

16 Cristiano Calcagno, Peter W. O'Hearn, and Hongseok Yang. Local action and abstract separation logic. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 366–378. IEEE Computer Society, 2007. `doi:10.1109/LICS.2007.30`.

17 Cristiano Calcagno, Hongseok Yang, and Peter W. O'Hearn. Computability and complexity results for a spatial assertion language for data structures. In Ramesh Hariharan, Madhavan Mukund, and V. Vinay, editors, *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science, 21st Conference, Bangalore, India, December 13-15, 2001, Proceedings*, volume 2245 of *Lecture Notes in Computer Science*, pages 108–119. Springer, 2001.

18 Luca Cardelli, Philippa Gardner, and Giorgio Ghelli. A Spatial Logic for Querying Graphs. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, editors, *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP'02)*, volume 2380 of *Lecture Notes in Computer Science*, pages 597–610. Springer, July 2002. `doi:10.1007/3-540-45465-9_51`.

19 Luca Cardelli and Andrew D. Gordon. Anytime, anywhere: Modal logics for mobile ambients. In *Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '00, pages 365–377, New York, NY, USA, 2000. Association for Computing Machinery. `doi:10.1145/325694.325742`.

20 Matthew Collinson, Kevin McDonald, and David J. Pym. A substructural logic for layered graphs. *J. Log. Comput.*, 24(4):953–988, 2014. doi:10.1093/logcom/exu002.

21 Byron Cook, Christoph Haase, Joël Ouaknine, Matthew J. Parkinson, and James Worrell. Tractable reasoning in a fragment of separation logic. In Joost-Pieter Katoen and Barbara König, editors, *CONCUR 2011 – Concurrency Theory – 22nd International Conference, CONCUR 2011, Aachen, Germany, September 6-9, 2011. Proceedings*, volume 6901 of *Lecture Notes in Computer Science*, pages 235–249. Springer, 2011. doi:10.1007/978-3-642-23217-6_16.

22 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

23 Bruno Courcelle. The monadic second-order logic of graphs VII: graphs as relational structures. *Theor. Comput. Sci.*, 101(1):3–33, 1992. doi:10.1016/0304-3975(92)90148-9.

24 Bruno Courcelle. Monadic second-order definable graph transductions: A survey. *Theor. Comput. Sci.*, 126(1):53–75, 1994. doi:10.1016/0304-3975(94)90268-2.

25 Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2012. doi:10.1017/CBO9780511977619.

26 Anuj Dawar, Philippa Gardner, and Giorgio Ghelli. Expressiveness and Complexity of Graph Logic. *Information and Computation*, 205(3):263–310, February 2007. doi:10.1016/j.ic.2006.10.006.

27 Pierpaolo Degano, Rocco De Nicola, and José Meseguer, editors. *Concurrency, Graphs and Models, Essays Dedicated to Ugo Montanari on the Occasion of His 65th Birthday*, volume 5065 of *Lecture Notes in Computer Science*. Springer, 2008. doi:10.1007/978-3-540-68679-8.

28 Stéphane Demri and Morgan Deters. Expressive completeness of separation logic with two variables and no separating conjunction. *ACM Trans. Comput. Log.*, 17(2):12, 2016.

29 Stéphane Demri, Étienne Lozes, and Alessio Mansutti. The effects of adding reachability predicates in quantifier-free separation logic. *ACM Trans. Comput. Log.*, 22(2):14:1–14:56, 2021. doi:10.1145/3448269.

30 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

31 Cezara Drăgoi, Constantin Enea, and Mihaela Sighireanu. Local shape analysis for overlaid data structures. In Francesco Logozzo and Manuel Fähndrich, editors, *Static Analysis*, pages 150–171, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

32 Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite model theory.* Perspectives in Mathematical Logic. Springer, 1995.

33 Mnacho Echenim, Radu Iosif, and Nicolas Peltier. The bernays-schönfinkel-ramsey class of separation logic with uninterpreted predicates. *ACM Trans. Comput. Log.*, 21(3):19:1–19:46, 2020.

34 Mnacho Echenim, Radu Iosif, and Nicolas Peltier. Decidable Entailments in Separation Logic with Inductive Definitions: Beyond Establishment. In Christel Baier and Jean Goubault-Larrecq, editors, *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2021.20.

35 Mnacho Echenim, Radu Iosif, and Nicolas Peltier. Unifying decidable entailments in separation logic with inductive definitions. In André Platzer and Geoff Sutcliffe, editors, *Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings*, volume 12699 of *Lecture Notes in Computer Science*, pages 183–199. Springer, 2021. doi:10.1007/978-3-030-79876-5_11.

36 Mnacho Echenim, Radu Iosif, and Nicolas Peltier. Entailment is undecidable for symbolic heap separation logic formulæ with non-established inductive rules. *Inf. Process. Lett.*, 173:106169, 2022. doi:10.1016/j.ipl.2021.106169.

37 Diego Figueira and Leonid Libkin. Path logics for querying graphs: Combining expressiveness and efficiency. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pages 329–340. IEEE Computer Society, 2015. doi:10.1109/LICS.2015.39.

38 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

39 S. Greibach. A note on undecidable properties of formal languages. *Math. Systems Theory*, 2:1–6, 1968. doi:10.1007/BF01691341.

40 Neil Immerman. *Second-Order Logic and Fagin's Theorem*, pages 113–124. Springer New York, New York, NY, 1999. doi:10.1007/978-1-4612-0539-5_8.

41 Radu Iosif, Adam Rogalewicz, and Jirí Simácek. The tree width of separation logic with recursive definitions. In Maria Paola Bonacina, editor, *Automated Deduction – CADE-24 – 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings*, volume 7898 of *Lecture Notes in Computer Science*, pages 21–38. Springer, 2013. doi:10.1007/978-3-642-38574-2_2.

42 Radu Iosif and Florian Zuleger. Expressiveness results for an inductive logic of separated relations, 2023. arXiv:2307.02381.

43 Samin S. Ishtiaq and Peter W. O'Hearn. BI as an assertion language for mutable data structures. In Chris Hankin and Dave Schmidt, editors, *Conference Record of POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, London, UK, January 17-19, 2001*, pages 14–26. ACM, 2001. doi:10.1145/360204.375719.

44 Christina Jansen, Jens Katelaan, Christoph Matheja, Thomas Noll, and Florian Zuleger. Unified reasoning about robustness properties of symbolic-heap separation logic. In *European Symposium on Programming (ESOP)*, volume 10201 of *Lecture Notes in Computer Science*, pages 611–638. Springer, 2017.

45 Neil D. Jones and Steven S. Muchnick. A flexible approach to interprocedural data flow analysis and programs with recursive data structures. In *Proceedings of the 9th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '82, pages 66–74, New York, NY, USA, 1982. Association for Computing Machinery. doi:10.1145/582153.582161.

46 Jens Katelaan, Dejan Jovanovic, and Georg Weissenbacher. A separation logic with data: Small models and automation. In Didier Galmiche, Stephan Schulz, and Roberto Sebastiani, editors, *Automated Reasoning – 9th International Joint Conference, IJCAR 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings*, volume 10900 of *Lecture Notes in Computer Science*, pages 455–471. Springer, 2018. doi:10.1007/978-3-319-94205-6_30.

47 Jens Katelaan and Florian Zuleger. Beyond symbolic heaps: Deciding separation logic with inductive definitions. In Elvira Albert and Laura Kovács, editors, *LPAR 2020: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Alicante, Spain, May 22-27, 2020*, volume 73 of *EPiC Series in Computing*, pages 390–408. EasyChair, 2020. doi:10.29007/vkmj.

48 Viktor Kuncak and Martin Rinard. Generalized records and spatial conjunction in role logic. In *Static Analysis*, pages 361–376, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

49 Viktor Kuncak and Martin C. Rinard. On spatial conjunction as second-order logic. *CoRR*, cs.LO/0410073, 2004. URL: http://arxiv.org/abs/cs.LO/0410073.

50 Étienne Lozes. *Expressivité des logiques spatiales*. Thèse de doctorat, Laboratoire de l'Informatique du Parallélisme, ENS Lyon, France, November 2004. URL: http://www.lsv.ens-cachan.fr/Publis/PAPERS/PS/PhD-lozes.ps.

51 Johann A. Makowsky. Algorithmic uses of the feferman-vaught theorem. *Ann. Pure Appl. Log.*, 126(1-3):159–213, 2004.

52 Alessio Mansutti. *Logiques de séparation : complexité, expressivité, calculs. (Reasoning with separation logics : complexity, expressive power, proof systems)*. PhD thesis, University of Paris-Saclay, France, 2020. URL: https://tel.archives-ouvertes.fr/tel-03094373.

53 Christoph Matheja, Jens Pagel, and Florian Zuleger. A decision procedure for guarded separation logic complete entailment checking for separation logic with inductive definitions. *ACM Trans. Comput. Log.*, 24(1):1:1–1:76, 2023. `doi:10.1145/3534927`.

54 Peter W. O'Hearn and David J. Pym. The logic of bunched implications. *Bull. Symb. Log.*, 5(2):215–244, 1999.

55 M. R. Fellows R. G. Downey. *Parameterized Complexity.* Springer New York, NY, 1999. `doi:10.1007/978-1-4612-0515-9`.

56 John C. Reynolds. Separation logic: A logic for shared mutable data structures. In *17th IEEE Symposium on Logic in Computer Science (LICS 2002), 22-25 July 2002, Copenhagen, Denmark, Proceedings*, pages 55–74. IEEE Computer Society, 2002. `doi:10.1109/LICS.2002.1029817`.

57 D. Seese. The structure of the models of decidable monadic theories of graphs. *Annals of Pure and Applied Logic*, 53(2):169–195, 1991. `doi:10.1016/0168-0072(91)90054-P`.

58 Dirk van Dalen. *Logic and structure (3. ed.).* Universitext. Springer, 1994.

59 Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 137–146. ACM, 1982.