Expressiveness Results for an Inductive Logic of Separated Relations

Radu Ioșif
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 formulae 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 [M]SO are incomparable on structures of unbounded treewidth, while SLR can be embedded in [M]SO in general. Furthermore, [M]SO 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 [M]SO 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


Funding Radu Ioșif: 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 ([M]SO), 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 [M]SO-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 formulae or validity of entailments between formulae.

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

\[(1) \text{ls}(x, y) \leftarrow x = y \quad \text{and} \quad (2) \text{ls}(x, y) \leftarrow \exists z . \ x \mapsto \rightarrow z \ast \text{ls}(z, y)\]

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 $\text{ls}(z, y)$. 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.

<table>
<thead>
<tr>
<th></th>
<th>SLR</th>
<th>MSO</th>
<th>SO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLR</td>
<td>✓</td>
<td>?</td>
<td>×</td>
</tr>
<tr>
<td>MSO</td>
<td>× (§4)</td>
<td>✓ (§6)</td>
<td>✓</td>
</tr>
<tr>
<td>SO</td>
<td>× (§4)</td>
<td>?</td>
<td>× (§7)</td>
</tr>
</tbody>
</table>

The SLR separating conjunction is understood as splitting the interpretation of each relation symbol from the signature into disjoint parts. For instance, the formula \( R(x_1, \ldots, x_n) \) describes a structure in which all relations are empty and \( R \) consists of a single tuple of values \( x_1, \ldots, x_n \), whereas \( R(x_1, \ldots, x_n) \star R(y_1, \ldots, y_n) \) 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, Gardiner 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 \( \mathcal{D} \) and a binary relation \( \mathcal{F} \), as (1) \( \text{rls}(x, y) \leftarrow x = y \) and (2) \( \text{rls}(x, y) \leftarrow \exists z : \mathcal{D}(x) \star \mathcal{F}(x, z) \star \text{rls}(z, y) \). Note that the \( \mathcal{D}(x) \) 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 \( \mathcal{D} \) for the definition of a Courcelle-style composition operator [23], where \( \mathcal{D} \) 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) \( \text{star}(x) \leftarrow \mathcal{M}(x) \star \text{node}(x) \) (2) \( \text{node}(x) \leftarrow x = x \) and (3) \( \text{node}(x) \leftarrow \exists y : \mathcal{E}(x, y) \star \mathcal{M}(y) \star \text{node}(x) \). 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 \( \mathcal{L} \in \{ \text{SLR, MSO, SO} \} \) using a finite set \( \Sigma \) 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 \( \mathcal{D} \) not in \( \Sigma \), 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 \( \mathcal{D} \). Then \( [\mathcal{L}]^{\mathcal{D}_k} \) 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 \( \mathcal{D} \), and \( [\mathcal{L}_1]^{\mathcal{D}_k} \subseteq [\mathcal{L}_2] \) 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 \( [\mathcal{L}]^{\mathcal{D}_k} \subseteq [\mathcal{L}] \) 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 \( [\mathcal{L}_1]^{\mathcal{D}_k} \subseteq [\mathcal{L}_2] \) (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. $\text{SO}$ is strictly more expressive than $\text{SLR}$, when considering unguarded structures of unbounded treewidth, and at least as expressive as $\text{SLR}$, when considering guarded structures of bounded treewidth.

3. $\text{SLR}$ is strictly more expressive than $\text{MSO}$, when considering guarded structures of bounded treewidth; this shows the expressive power of $\text{SLR}$, emphasizing (once more) the model-theoretic importance of the treewidth parameter.

Note that, when considering $\text{SLR}$-definable sets of bounded treewidth, we systematically assume these structures to be guarded. We state as an open problem and conjecture that every infinite $\text{SLR}$-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 $\text{SL}$ symbolic heaps [41, 47] and of invariance for assertions written in a fragment of $\text{SLR}$ 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 $\text{SLR}$ with the same expressive power as $\text{MSO}$, over structures of bounded treewidth. This is also motivated by the need for a general fragment of $\text{SLR}$ with a decidable entailment problem, that is instrumental in designing automated verification systems. Unfortunately, such a definition is challenging because the $\text{MSO}$-definability of the sets defined by $\text{SLR}$ 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 $\text{NP}$-complete graph problems such as Hamiltonicity and $\text{3}$-Coloring become $\text{PTIME}$, 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 ($\text{MSO}$) logical theories. A result of Courcelle [22] proves that $\text{MSO}$ 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 $\text{MSO}$ theory necessarily has bounded treewidth.

Comparing the expressiveness of $\text{SL}$ [56] with classical logics received a fair amount of attention. A first proof of undecidability of the satisfiability problem for $\text{SL}$, 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 $R(x, y) \overset{\text{def}}{=} \exists z . z \rightarrow (x, y) \rightarrow \text{true}$. A more refined proof for heaps of outdegree one was given in [11], where it was shown that $\text{SO}$ has the same expressivity as $\text{SL}$, when negation and separating implication is allowed, which is not the case for our fragment of $\text{SLR}$.

A related line of work, pioneered by Lozes [50], is the translation of quantifier-free $\text{SL}$ 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 $\text{SL}$ 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 $\text{SL}$ into first order logic, based on the small model property of the former, has been described in [15]. These are fragments of $\text{SL}$ 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 \( \text{MSO}_2 \), i.e., \( \text{MSO} \) 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 \( \text{MSO}_2 \) 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 \( \text{MSO}_2 \) are simple graphs. Without recursion, GL can be translated into \( \text{MSO}_2 \) and it has been shown that \( \text{MSO}_2 \) is strictly more expressive than GL without edge label quantification [5]. Little is known for GL with recursion, besides that it can express \( \text{PSPACE} \)-complete model checking problems [26], whereas model checking is \( \text{PSPACE} \)-complete for \( \text{MSO} \) [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 developed 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 \( \pow(A) \) its powerset, \( A^1 \overset{\text{def}}{=} A, A^{i+1} \overset{\text{def}}{=} A^i \times A \), for all \( i \geq 1 \), where \( \times \) is the Cartesian product, and \( A^+ \overset{\text{def}}{=} \bigcup_{i \geq 1} A^i \). 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 \( \dom(f) \) its domain and by \( f\restriction_s \) its restriction to \( S \subseteq \dom(f) \). \( f \) 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 \( a \in \dom(f) \), \( \mathcal{M} \) outputs \( f(a) \) in finitely many steps and diverges for \( a \notin \dom(f) \).

### Signatures and Structures.

Let \( \Sigma = \{R_1, \ldots, R_N, c_1, \ldots, c_M\} \) be a finite signature, where \( R_i \) are relation symbols of arity \( \#R_i \geq 1 \) 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 \( \Sigma \). Unless stated otherwise, we consider \( \Sigma \) and \( \mathfrak{D} \) to be fixed in the following.

A structure is a pair \((U, \sigma)\), where \( U \) is an infinite set, called universe, and \( \sigma : \Sigma \to U \cup \pow(U^+) \) is an interpretation that maps each relation symbol \( R \) to a relation \( \sigma(R) \subseteq U^{\#R} \) and each constant \( c \) to an element \( \sigma(c) \in U \). 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 \( \text{Rel}(\sigma) \) for the set of elements that belong to \( \sigma(R) \), for some relation symbol \( R \in \Sigma \) and \( \text{Supp}(\sigma) \overset{\text{def}}{=} \text{Rel}(\sigma) \cup \{\sigma(c_1), \ldots, \sigma(c_M)\} \) for the support of the structure, that includes the interpretation of constants. We denote by \( \text{Str}(\Sigma) \) (resp. \( \text{Str}(\Sigma, \mathfrak{D}) \)) the set of structures over the signature \( \Sigma \) (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 $\mathcal{D}$:

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

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

**Definition 2.** The composition of two compatible and locally disjoint structures $(U_1, \sigma_1)$ and $(U_2, \sigma_2)$ is $(U_1, \sigma_1) \cdot (U_2, \sigma_2) \overset{\Delta}{=} (U_1 \cup U_2, \sigma_1 \uplus \sigma_2)$, where $(\sigma_1 \uplus \sigma_2)(R) \overset{\Delta}{=} \sigma_1(R_1) \cup \sigma_2(R_2)$ and $(\sigma_1 \uplus \sigma_2)(c_j) \overset{\Delta}{=} \sigma_1(c_j) = \sigma_2(c_j)$, 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 $G = (V, E)$, such that $V$ is a set of vertices and $E \subseteq V \times V$ is a set of edges. All graphs considered in this paper are finite and directed, i.e., $E$ is not necessarily a symmetric relation. Graphs are naturally encoded as structures:

**Definition 3.** A graph $G = (V, E)$ is encoded by the structure $(U_G, \sigma_G)$ over the signature $\Gamma \overset{\Delta}{=} \{ \mathcal{U}, \mathcal{E} \}$, such that $\#\mathcal{U} = 1$ and $\#\mathcal{E} = 2$, such that $U_G = V$, $\sigma_G(\mathcal{U}) = V$ and $\sigma_G(\mathcal{E}) = E$.

A path in $G$ is a sequence of pairwise distinct vertices $v_1, \ldots, v_n$, such that $(v_i, v_{i+1}) \in E$, for all $i \in [1, n - 1]$. We say that $v_1, \ldots, v_n$ is an undirected path if $\{(v_i, v_{i+1}), (v_{i+1}, v_i)\} \cap E \neq \emptyset$ instead, for all $i \in [1, n - 1]$. A set of vertices $V \subseteq V$ is connected in $G$ iff there is an undirected path in $G$ between any two vertices in $V$. A graph $G$ is connected iff $V$ is connected in $G$. 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 $K_n$ the set of cliques with $n$ vertices.

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

**Definition 4.** A tree decomposition of a structure $(U, \sigma)$ over the signature $\Sigma$ is a pow($U$)-labeled tree $T = (N, F, r, \lambda)$, such that the following hold:

1. for each relation symbol $R \in \Sigma$ and each tuple $(u_{i,1}, \ldots, u_{i,n}) \in \sigma(R)$ there exists $n \in N$, such that $\{u_{i,1}, \ldots, u_{i,n}\} \subseteq \lambda(n)$, and
2. for each $u \in \text{Supp}(\sigma)$, the set $\{n \in N \mid u \in \lambda(n)\}$ is nonempty and connected in $(N, F)$.

The width of the tree decomposition is $\text{tw}(T) \overset{\Delta}{=} \max_{n \in N} |\lambda(n)| - 1$. The treewidth of the structure $(U, \sigma)$ is $\text{tw}(U, \sigma) \overset{\Delta}{=} \min\{\text{tw}(T) \mid T$ 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 $\{K_n \mid n \in \mathbb{N}\}$ is strictly treewidth-unbounded.
3 Logics

We introduce two logics over a relational signature $\Sigma = \{R_1, \ldots, R_N, c_1, \ldots, c_M\}$. First, the Separation Logic of Relations (SLR) uses a set of first order variables $V_1 = \{x, \ldots\}$ and a set of predicates $\mathcal{A} = \{A, \ldots\}$ (also called recursion variables in the literature, e.g., [18]) of arities $\#A \geq 0$. We use the symbols $\xi, \chi \in V_1 \cup \{c_1, \ldots, c_M\}$ to denote terms, i.e., either first order variables or constants. The formulæ of SLR are defined by the following syntax:

$$\phi := \text{emp} | \chi | \xi \neq \chi | R(\xi_1, \ldots, \xi_R), A(\xi_1, \ldots, \xi_A), \phi \star \phi | \exists x . \phi$$

The formulæ $\xi = \chi$ and $\xi \neq \chi$ are called equalities and disequalities, $R(\xi_1, \ldots, \xi_R)$ and $A(\xi_1, \ldots, \xi_A)$ 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 $fv(\phi)$ be the set of free variables of $\phi$. 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 $\xi_i$ in $\phi$, for all $i \in [1, n]$.

As a convention, the bound variables in $\phi$ are renamed to avoid clashes with $\xi_1, \ldots, \xi_n$.

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

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

The semantics of SLR formulæ is given by the satisfaction relation $(U, \sigma) \models^\Delta \phi$ between structures and formulæ. This relation is parameterized by a store $\nu : V_1 \rightarrow U$ mapping the free variables of a formula into elements of the universe and an SID $\Delta$. We write $\nu[x \leftarrow u]$ for the store that maps $x$ into $u$ and agrees with $\nu$ on all variables other than $x$. For a term $\xi$, we denote by $(\sigma, \nu)(\xi)$ the value $\sigma(\xi)$ if $\xi$ is a constant, or $\nu(\xi)$ if $\xi$ is a first-order variable.

The satisfaction relation is the least relation that satisfies the following conditions:

\[
\begin{align*}
(U, \sigma) \models^\Delta \text{emp} & \iff \sigma(R) = \emptyset, \text{ for all } R \in \Sigma \\
(U, \sigma) \models^\Delta \xi \sim \chi & \iff (U, \sigma) \models^\Delta \text{emp} \text{ and } (\sigma, \nu)(\xi) \sim (\sigma, \nu)(\chi), \text{ where } \sim \in \{=, \neq\} \\
(U, \sigma) \models^\Delta R(\xi_1, \ldots, \xi_R) & \iff \sigma(R) = \{(\sigma, \nu)(\xi_1), \ldots, (\sigma, \nu)(\xi_R)\} \\
& \quad \text{ and } \sigma(R') = \emptyset, \text{ for } R' \in \Sigma \setminus \{R\} \\
(U, \sigma) \models^\Delta A(\xi_1, \ldots, \xi_A) & \iff (U, \sigma) \models^\Delta \phi[x_1/\xi_1, \ldots, x_A/\xi_A], \text{ for some } A(x_1, \ldots, x_A) \leftarrow \phi \in \Delta \\
(U, \sigma) \models^\Delta \phi_1 \star \phi_2 & \iff \text{there exist structures } (U_1, \sigma_1) \text{ and } (U_2, \sigma_2), \text{ such that } \\
& \quad (U, \sigma) = (U_1, \sigma_1) \bullet (U_2, \sigma_2) \text{ and } (U, \sigma_i) \models^\Delta \phi_i, \text{ for } i = 1, 2 \\
(U, \sigma) \models^\Delta \exists x . \phi & \iff (U, \sigma) \models^\nu[\nu[x \leftarrow a] \phi, \text{ for some } a \in U \\
\end{align*}
\]

Note that every structure $(U, \sigma)$, such that $(U, \sigma) \models^\nu \phi$, interprets each relation symbol as a finite set of tuples, defined by a finite least fixpoint iteration over the rules in $\Delta$. 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 $\phi$ 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 $(U, \sigma)$ is a $\Delta$-model of $\phi$. We denote by $[\phi]^\Delta$ the set of $\Delta$-models of $\phi$. We call $[\phi]^\Delta$ an SLR-definable set. By $[\phi]^{D,k}_\Delta$ we denote the set of guarded structures (Def. 1) of treewidth at most $k$ from $[\phi]^\Delta$. We write $[\mathcal{S}L] \equiv \{[\phi]^\Delta | \phi \text{ is a SLR formula, } \Delta \text{ is a SID}\}$ and $[\mathcal{S}L]^{D,k} \equiv \{[\phi]^{D,k}_\Delta | \phi \text{ is a SLR formula, } \Delta \text{ is a SID}\}$.

Below we show that SLR-definable sets are unions of isomorphic equivalence classes:
Proposition 7. Given isomorphic structures \((U, \sigma)\) and \((U', \sigma')\), for any sentence \(\phi\) of SLR and any SIID \(\Delta\), we have \((U, \sigma) \models \Delta \iff (U', \sigma') \models \Delta \phi\).

The other logic is the Weak Second Order Logic (SO) defined using a set of second order variables \(V_2 = \{X, \ldots\}\), in addition to first order variables \(V_1\). We denote by \(#X\) the arity of a second order variable \(X\). Terms and atoms are the same as in SLR. The formulae of SO have the following syntax:

\[
\psi := \xi = \chi \mid R(\xi_1, \ldots, \xi_{\#R}) \mid X(\xi_1, \ldots, \xi_{\#X}) \mid \neg \psi \mid \psi \land \psi \mid \exists x . \psi \mid \exists X . \psi
\]

We write \(\xi \not= \chi \overset{def}{=} \neg \xi = \chi\), \(\psi_1 \lor \psi_2 \overset{def}{=} \neg(\neg \psi_1 \land \neg \psi_2)\), \(\psi_1 \rightarrow \psi_2 \overset{def}{=} \neg \psi_1 \lor \psi_2\), \(\forall x . \psi \overset{def}{=} \exists x . \neg \psi\), and \(\forall X . \psi \overset{def}{=} \neg \exists X . \neg \psi\). 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 formulae of the form \(\exists X_1 \ldots \exists X_n . \phi\), where \(\phi\) has only first order quantifiers.

The semantics of SO is given by a relation \((U, \sigma) \models^\nu \psi\), where the store \(\nu : V_1 \cup V_2 \rightarrow U \cup \text{pow}(U^+)\) maps each first-order variable \(x \in V_1\) to an element of the universe \(\nu(x) \in U\) and each second-order variable \(X \in V_2\) to a finite relation \(\nu(X) \subseteq U^{#X}\). The satisfaction relation of SO is defined inductively on the structure of formulae:

\[
\begin{align*}
(U, \sigma) \models^\nu \xi = \chi & \iff (\sigma, \nu) (\xi) = (\sigma, \nu) (\chi) \\
(U, \sigma) \models^\nu R(\xi_1, \ldots, \xi_{\#R}) & \iff ((\sigma, \nu) (\xi_1), \ldots, (\sigma, \nu) (\xi_{\#R})) \in \sigma (R) \\
(U, \sigma) \models^\nu X(\xi_1, \ldots, \xi_{\#X}) & \iff ((\sigma, \nu) (\xi_1), \ldots, (\sigma, \nu) (\xi_{\#X})) \in \nu (X) \\
(U, \sigma) \models^\nu \exists X . \psi & \iff (U, \sigma) \models^{\nu [X \mapsto V]} \psi, \text{ for some finite set } V \subseteq U^{#X}
\end{align*}
\]

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 \(\phi\) is a sentence, we write \((U, \sigma) \models^\nu \phi\) instead of \((U, \sigma) \models^m \phi\) and define \(\models^\nu \phi\overset{def}{=} \{(U, \sigma) \mid (U, \sigma) \models^\nu \phi\}\) and \(\models^{D,k} \phi\) for the restriction of \(\models^\nu \phi\) to guarded structures of treewidth at most \(k\). We call \(\models^\nu \phi\) an (MSO)-definable set. We write \([MSO]^\nu \equiv \{\models^\nu \mid \phi\text{ is a (MSO) formula}\}\) and \([MSO]^{D,k} \equiv \{\models^{D,k} \phi \mid \phi\text{ is a (MSO) formula}\}\).

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 \([L_1] \subseteq [L_2]\) and \([L_1]^{D,k} \subseteq [L_2]\), where \(L_1\) and \(L_2\) are any of the logics SLR, MSO and SO, respectively. In particular, for \([L_1]^{D,k} \subseteq [L_2]\), we implicitly assume that \(L_1\) and \(L_2\) are sets of formulæ over the relational signature \(\Sigma \cup \{\Delta\}\). 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.

\[\text{SLR}^{D,k} \not\subseteq [MSO] \not\subseteq [SLR]\]

The argument that shows \([SLR]^{D,k} \not\subseteq [MSO]\) 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 \(\Delta\) of arity zero:
\begin{align*}
A() & \leftarrow \exists x \exists y . \mathfrak{R}(x) \ast \mathfrak{R}(y) \ast A() \quad A() \leftarrow \text{emp}
\end{align*}

Note that every model of 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 \([\text{SLR}]^D \not\subseteq [\text{MSO}]^k\) for any \( k \geq 1 \), and we obtain \([\text{SLR}] \not\subseteq [\text{MSO}]\), in general.

The argument for \([\text{MSO}] \not\subseteq [\text{SLR}]\) 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):

\[
\forall x \forall y . \mathfrak{M}(x) \land \mathfrak{M}(y) \land x \neq y \rightarrow \mathfrak{E}(x, y) \lor \mathfrak{E}(y, x)
\]

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 \( \phi \) and SID \( \Delta \), we prove the existence of an integer \( W \geq 1 \), depending on \( \phi \) and \( \Delta \) alone, such that

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

(ii) the function that maps \( (U, \sigma) \) into \( (U', \sigma') \) 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:

\begin{definition}
A rule \( A(x_1, \ldots, x_{\#A}) \leftarrow \exists y_1 \ldots \exists y_n . \psi \), where \( \psi \) is a quantifier-free formula, is normalized if no equality atom \( x = y \) occurs in \( \psi \), for distinct variables \( x, y \in \{x_1, \ldots, x_{\#A}\} \cup \{y_1, \ldots, y_n\} \). An SID is normalized if it contains only normalized rules.
\end{definition}

\begin{lemma}
Given an SID \( \Delta \), one can build a normalized SID \( \Delta' \) such that, for each structure \( \sigma \) and each predicate atom \( A(\xi_1, \ldots, \xi_{\#A}) \), we have \( (U, \sigma) \models_\Delta \exists \xi_1 \ldots \exists \xi_n . A(\xi_1, \ldots, \xi_{\#A}) \iff (U, \sigma) \models_{\Delta'} \exists \xi_1 \ldots \exists \xi_n . A(\xi_1, \ldots, \xi_{\#A}) \), where \( \{\xi_1, \ldots, \xi_n\} = \{\xi_1, \ldots, \xi_{\#A}\} \cap \forall_1 \).
\end{lemma}

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 \( \text{fold}_\text{ls}(x_1) \leftarrow \text{emp} \) and \( \text{fold}_\text{ls}(x_1) \leftarrow \exists y . \delta(x_1, y) \ast \text{fold}_\text{ls}(y) \), the \( \text{fold}_\text{ls}(x) \) formula defines an infinite set of graphs whose edges are given by the interpretation of a relation symbol \( \mathfrak{R} \), 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 \( \text{fold}_\text{ls}(x) \) 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 \( (U, \sigma) \models_{\Delta}^\ast \phi \) iff the satisfaction relation \( (U, \sigma) \models_{\Delta}^\ast \phi \) can be established by considering finite injective stores. The definition of \( \models_{\Delta}^\ast \) is the same as the one of \( \models_{\Delta}^\nu \) (\S3), except for the cases below:

\[
(U, \sigma) \models_{\Delta}^\ast \phi_1 \ast \phi_2 \iff \text{there exist structures } (U_1, \sigma_1) \ast (U_2, \sigma_2) = (U, \sigma), \text{ such that } U_1 \cap U_2 = \nu(\text{fv}(\phi_1) \cap \text{fv}(\phi_2)) \text{ and } (U, \sigma_i) \models_{\Delta}^\nu_{\nu(\text{fv}(\phi_i))} \phi_i, \text{ for } i = 1, 2
\]

\[
(U, \sigma) \models_{\Delta}^\ast \exists x . \phi \iff (U, \sigma) \models_{\Delta}^\nu_{|\nu(\text{fv}(\phi))|} \phi, \text{ for some } u \in U \setminus \nu(\text{fv}(\phi))
\]

For instance, we have \( (U, \sigma) \models_{\Delta}^\ast \text{fold}_\text{ls}(x) \) only if \( \sigma(\emptyset) \) is a list of pairwise distinct elements.
Lemma 10. Given a normalized SID $\Delta$, a predicate atom $A(\xi_1, \ldots, \xi_{\#A})$, for each structure $(U, \sigma)$ and a store $\nu$, such that $(U, \sigma) \models^\Delta A(\xi_1, \ldots, \xi_{\#A})$, there exists a structure $(U, \pi)$, such that $(U, \pi) \models^\Delta A(\xi_1, \ldots, \xi_{\#A})$. Moreover, the function with domain $[A(\xi_1, \ldots, \xi_{\#A})]^{\Delta}$ that maps $(U, \sigma)$ into the set of structures isomorphic with $(U, \pi)$ is locally co-finite.

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

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

Note that proving Lemmas 10 and 11 for predicate atoms loses no generality, because for each formula $\phi$, such that $\text{fv}(\phi) = \{x_1, \ldots, x_n\}$, we can consider a predicate symbol $A_{\phi}$ of arity $n$ and extend the SID by the rule $A_{\phi}(x_1, \ldots, x_n) \leftarrow \phi$. The proof of $[\text{MSO}] \subsetneq [\text{SLR}]$ relies on the following:

Proposition 12. Given a sentence $\phi$ and an SID $\Delta$, $[\phi]^\Delta$ is either finite or it has an infinite subset of bounded treewidth.

5 \hspace{1cm} [\text{SLR}] \subseteq [\text{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 $[\text{MSO}] \subseteq [\text{SO}]$ trivially. We show that $[\text{SLR}] \subseteq [\text{SO}]$ 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 $\Delta \overset{\text{def}}{=} \{t_1, \ldots, t_R\}$ be a given SID. Without loss of generality, for each relation symbol $R \in \Sigma$, we assume that there is at most one occurrence of an atom $R(y_1, \ldots, y_{\#R})$ in each rule from $\Delta$. If this is not the case, we split the rule by introducing a new predicate symbol for each relation atom with relation symbol $R$, until the condition is satisfied.

Definition 13. An unfolding tree for a predicate atom $A(\xi_1, \ldots, \xi_{\#A})$ is a $\Delta$-labeled tree $T = (\mathcal{N}, \mathcal{F}, r, \lambda)$, such that $\lambda(r)$ defines $A$ and, for each vertex $n \in \mathcal{N}$, if $B_1(z_{1,1}, \ldots, z_{1,\#B_1}), \ldots, B_h(z_{h,1}, \ldots, z_{h,\#B_h})$ are the predicate atoms that occur in $\lambda(n)$, then $p_1, \ldots, p_h$ are the children of $n$ in $T$, such that $\lambda(p_{\ell})$ defines $B_{\ell}$, for all $\ell \in [1, h]$.

We build a SO formula that defines the models of a relation atom $A(\xi_1, \ldots, \xi_{\#A})$. As explained above, this is without loss of generality. Let $P$ be the maximum number of occurrences of predicate atoms in a rule from $\Delta$. 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 $\Delta$. First, we build a SO formula $\mathcal{T} = \{X_j \mid j = 1, \ldots, R\}$, as the conjunction of SO formulae 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 $X_{j_k}$, respectively, such that $r_{j_k}$ defines $B_{\ell_k}$ for all $\ell_k \in [1, h]$, where $B_{1}(z_1, \ldots, z_1, b_{1,1}), \ldots, B_{h}(z_h, \ldots, z_h, b_{h,1})$ are the predicate atoms that occur in $r_i$.

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

(i) each second order variable $Z_{k, \ell, t}$ denotes a functional binary relation,
(ii) for each tree node labeled by a rule $r_i$ and each atom $R_k(\xi_1, \ldots, \xi_{\#R_k})$ occurring at that node, the interpretation of $R_k$ contains a tuple, whose elements are related to the node via $Z_{k, 1, 1}, \ldots, Z_{k, 1, 1}$, 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 $\xi_m$ and $\chi_n$ that occur in two relation atoms $R_k(\xi_1, \ldots, \xi_{\#R_k})$ and $R_{\ell}(\chi_1, \ldots, \chi_{\#R_{\ell}})$ 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 $R_k(\xi_1, \ldots, \xi_{\#R_k})$ and $R_{\ell}(\chi_1, \ldots, \chi_{\#R_{\ell}})$ in rules $r_{j_k}$ and $r_{j_{\ell}}$, respectively, such that $\xi$ is bound by $\varsigma$ and $\chi$ to $\varsigma_s$ by equalities and parameter passing,
(vii) each term in $A(\xi_1, \ldots, \xi_{\#A})$ that is bound to a variable from a relation atom $R_k(z_1, \ldots, z_{\#R_k})$ in the unfolding, must be equated to that variable.

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

$$\mathfrak{F}(\xi_1, \ldots, \xi_{\#A}) = \exists x \exists \{X_i\}_{i=1}^R \exists \{Y_j\}_{j=1}^P \exists \{Z_k, \ell, t\}_{\ell=1}^N \exists \{Z_k, \ell, t\}_{\ell=1}^N .$$

\[ \mathfrak{F}(x, [X_i]_{i=1}^R, [Y_j]_{j=1}^P) \land \mathfrak{F}(\xi_1, \ldots, \xi_{\#A}, x, [X_i]_{i=1}^R, [Y_j]_{j=1}^P, \{\{Z_k, \ell, t\}_{\ell=1}^N\}_{k=1}^N, \{\{Z_k, \ell, t\}_{\ell=1}^N\}_{k=1}^N) \]

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

**Proposition 14.** Given an SID $\Delta$ and a predicate atom $A(\xi_1, \ldots, \xi_{\#A})$, for each structure $(U, \sigma)$ and store $\nu$, we have $(U, \sigma) \models^{\nu} A(\xi_1, \ldots, \xi_{\#A})$ if and only if $(U, \sigma) \models^{\nu} \mathfrak{F}(\xi_1, \ldots, \xi_{\#A})$.

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 formulae are given in the full version of the paper [42].
We recall first a result of Courcelle [23], that describes the structures of bounded treewidth, which satisfy a given MSO formula \( \phi \), by an effectively constructible set of recursive equations. This set of equations uses two operations on structures, namely glue \( \text{glue} \) and \( \text{fgcst}_j \), 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 \( \phi \). Because this result applies to general (i.e., finite and infinite) structures \( (U, \sigma) \), 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 \( \phi \) in the rest of this section.

**Operations on Structures.** Let \( \Sigma_1 \) and \( \Sigma_2 \) be two (possibly overlapping) signatures. The glueing operation \( \text{glue} : \text{Str}(\Sigma_1) \times \text{Str}(\Sigma_2) \to \text{Str}(\Sigma_1 \cup \Sigma_2) \) 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 \( \sigma_1(c) \sim \sigma_2(c) \), for all \( c \in \Sigma_1 \cap \Sigma_2 \). 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 
\[
\text{glue}(S_1, S_2) \equiv (U, \sigma), \text{ where } U \equiv \{ [u] \mid u \in U_1 \cup U_2 \} \text{ and } \sigma \text{ is defined as follows:}
\]
\[
\sigma(R) \equiv \begin{cases} 
[\sigma_1(R)], & \text{if } R \in \Sigma_i \setminus \Sigma_{3-i}, \text{ for both } i = 1, 2 \\
[\sigma_1(R) \cup \sigma_2(R)], & \text{if } R \in \Sigma_1 \cap \Sigma_2 
\end{cases}
\]
Since we match isomorphic structures, the nature of the elements of \( U \) (i.e., equivalence classes) is not important. The forget operation \( \text{fgcst}_j : \text{Str}(\Sigma) \to \text{Str}(\Sigma \setminus \{c_j\}) \) simply drops the constant \( c_j \) from the domain of its argument.

**Structures of Bounded Treewidth.** Let \( \Sigma = \{R_1, \ldots, R_N, c_1, \ldots, c_M\} \) be a signature and \( \Pi = \{c_{M+1}, \ldots, c_{M+k+1}\} \) be a set of constants disjoint from \( \Sigma \), 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 \( Tw(k) \) is the set of recursive equations of the form

\[
Y_0 \supseteq f(Y_1, \ldots, Y_n),
\]
where each \( f \) is either \( \text{glue}, \text{fgcst}_{M+j} \), 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 \( Tw(k) \), in the domain of tuples of sets ordered by pointwise inclusion [25, Theorem 2.83].
Models of MSO Formulae. The quantifier rank $qr(\phi)$ of an MSO formula $\phi$ is the maximal depth of nested quantifiers, i.e., $qr(\phi) \equiv 0$ if $\phi$ is an atom, $qr(\neg \phi_1) \equiv qr(\phi_1)$, $qr(\phi_1 \land \phi_2) \equiv \max(qr(\phi_1), qr(\phi_2))$ and $qr(\exists x \cdot \phi_1) = qr(\exists X \cdot \phi_1) \equiv qr(\phi_1) + 1$. We denote by $F_{\text{MSO}}^k$ the set of MSO sentences of quantifier rank at most $k$. This set is finite, up to logical equivalence.

For a structure $S = (U, \sigma)$, we define its $r$-type as $\text{type}^r(S) \equiv \{ \phi \in F_{\text{MSO}}^k \mid S \vDash \phi \}$. We assume the sentences in $\text{type}^r(S)$ to use the signature over which $S$ is defined; this signature will be clear from the context in the following.

Definition 15. An operation $f : \text{Str}(\Sigma_1) \times \ldots \times \text{Str}(\Sigma_n) \to \text{Str}(\Sigma_{n+1})$ is (effectively) MSO-compatible\(^2\) iff, for all structures $S_1, \ldots, S_n$, $\text{type}^r(f(S_1, \ldots, S_n))$ depends only on (and can be effectively computed from) $\text{type}^r(S_1), \ldots, \text{type}^r(S_n)$ by an abstract operation $f^\sharp : (\text{pow}(F_{\text{MSO}}^r))^n \to \text{pow}(F_{\text{MSO}}^r)$.

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

6.2 Encoding Types in SLR

We begin explaining the proof for $[\text{MSO}]^{\mathbb{D}, k} \subseteq [\text{SLR}]$. Instead of using the set of recursive equations $Tw(k)$ 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 $\mathbb{D}$ is used to enforce disjointness of the arguments of the composition operation, in all but finitely many elements. Intuitively, $\mathbb{D}$ “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 $(U, \sigma) \in \text{Str}(\Sigma, \mathbb{D})$ with at least $k + 1$ distinct elements in $\sigma(\mathbb{D})$ (rule 4) and $\sigma(\mathbb{R}) \neq \emptyset$ for at least one relation symbol $\mathbb{R} \in \Sigma$ (rule 3). The cases of structures such that $|\sigma(\mathbb{D})| \leq k$ or $\bigcup_{\mathbb{R} \in \Sigma} \sigma(\mathbb{R}) = \emptyset$ 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 $(U, \sigma) \in \text{Str}(\Sigma, \mathbb{D})$, such that $|\sigma(\mathbb{D})| \geq k + 1$ and $\sigma(\mathbb{R}) \neq \emptyset$, for at least some $\mathbb{R} \in \Sigma$, we have $\text{tw}(\sigma) \leq k$ iff $(U, \sigma) \models_{\Delta(k)} A_k(\cdot)$.

\(^2\) Also referred to as smooth operations in [51].
\[ A(x_1, \ldots, x_{k+1}) \leftarrow A(x_1, \ldots, x_{k+1}) \ast A(x_1, \ldots, x_{k+1}) \]  
\[ A(x_1, \ldots, x_{k+1}) \leftarrow \exists y. \mathcal{D}(y) \ast A(x_1, \ldots, x_{k+1})[x_i/y] \text{ for all } i \in [1, k+1] \]  
\[ A(x_1, \ldots, x_{k+1}) \leftarrow R(y_1, \ldots, y_{\#R}) \text{ for all } R \in \Sigma \text{ and } y_1, \ldots, y_{\#R} \in \{x_1, \ldots, x_{k+1}\} \]  
\[ A_k() \leftarrow \exists x_1 \ldots \exists x_{k+1}. \mathcal{D}(x_1) \ast * \mathcal{D}(x_{k+1}) \ast A(x_1, \ldots, x_{k+1}) \]  

(a)  

\[ A^\tau(x_1, \ldots, x_{k+1}) \leftarrow A^\tau_0(x_1, \ldots, x_{k+1}) \ast A^\tau_2(x_1, \ldots, x_{k+1}) \text{ where } \tau = \text{glue}^2(\tau_1, \tau_2) \]  
\[ A^\tau(x_1, \ldots, x_{k+1}) \leftarrow \exists y. \mathcal{D}(y) \ast A^\tau_0(x_1, \ldots, x_{k+1})[x_i/y] \text{ for all } i \in [1, k+1], \text{ where } \tau = \text{glue}^2(\text{fgest}^*_M(\tau_1), \rho_1) \text{ and } \rho_1 \text{ is the type of some structure } S \in \text{Str}(\{c_{M+1}\}, \mathcal{D}) \text{ with singleton universe and } S \models \mathcal{D}(c_{M+1}) \]  
\[ A^\tau(x_1, \ldots, x_{k+1}) \leftarrow R(y_1, \ldots, y_{\#R}) \text{ for some } y_1, \ldots, y_{\#R} \in \{x_1, \ldots, x_{k+1}\}, \text{ where } \tau = \text{type}^{qr}(\phi), S \in \text{Str}(\Sigma \cup \{c_{M+1}, \ldots, c_{M+k+1}\}, \mathcal{D}) \text{ and } S \models R(y_1, \ldots, y_{\#R})[x_1/c_{M+1}, \ldots, x_{k+1}/c_{M+k+1}] \ast \ast \prod_{i=1}^{k+1} \mathcal{D}(c_{M+i}) \]  
\[ A_{k, \phi}() \leftarrow \exists x_1 \ldots \exists x_{k+1}. \mathcal{D}(x_1) \ast * \mathcal{D}(x_{k+1}) \ast A^\tau(x_1, \ldots, x_{k+1}) \text{ for all } \tau \text{ such that } \phi \in \tau \]  

(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 $\phi$, of treewidth at most $k$ (b).

We remark that the encoding of **glue** and **fgest** 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 **fgest**, 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 qr(ϕ)-types, in order to obtain an SID $\Delta(k, \phi)$ (Fig. 1b) describing the models of an MSO sentence $\phi$, of treewidth at most $k$. We consider the set of ports $\Pi = \{c_{M+1}, \ldots, c_{M+k+1}\}$ disjoint from $\Sigma$. 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 $\Sigma = \{R_1, \ldots, R_N, c_1, \ldots, c_M\}$ be a signature, $\Pi = \{c_{M+1}, \ldots, c_{M+k+1}\}$ be a set of constants not in $\Sigma$, and let $(U, \sigma) \in \text{Str}(\Sigma, \mathcal{D})$ be a structure. Let $\nu$ be a store mapping $x_1, \ldots, x_{k+1}$ to elements of $U \setminus \sigma(\mathcal{D})$. Then, $\text{encode}((U, \sigma), \nu) \in \text{Str}(\Sigma \cup \Pi, \mathcal{D})$ is a structure with universe $U$ that agrees with $(U, \sigma)$ over $\Sigma$, maps each $c_{M+i}$ to $\nu(x_i)$, for $i \in [1, k+1]$ and maps $\mathcal{D}$ to $\sigma(\mathcal{D}) \cup \{\nu(x_1), \ldots, \nu(x_{k+1})\}$.

The correctness of our construction relies on the fact that the composition acts like gluing, 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 $\Pi$:

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

$$\text{type}^e(\text{encode}((U_1, \sigma_1) \bullet (U_2, \sigma_2), \nu)) = \text{glue}^e(\text{type}^e(\text{encode}((U_1, \sigma_1), \nu)), \text{type}^e(\text{encode}((U_2, \sigma_2), \nu)))$$
Finally, the main property of $\Delta(k, \phi)$ is stated and proved below:

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

1. $(U, \sigma) \models \phi$ and $\text{tw}(\sigma) \leq k$, and
2. $(U, \sigma) \models \Delta(k, \phi) A_{k, \phi}(\cdot)$.

The above result shows that SLR can define the guarded models $(U, \sigma) \in \text{Str}(\Sigma, \mathcal{D})$ 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 $S = (U, \sigma) \in \text{Str}(\Sigma)$, we define the finite structure $S^n = (\text{Supp}(\sigma) \cup \{v_1, \ldots, v_n\}, \sigma)$, for pairwise distinct elements $v_1, \ldots, v_n \in U \setminus \text{Supp}(\sigma)$. Then, for any quantifier rank $r$, the structures $S$ and $S^{2r}$ have the same $r$-type:

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

As a final remark, we notice that the idea used to prove $[\text{MSO}]_{\mathcal{D},k} \subset [\text{SLR}]$ can be extended to show also $[\text{CMSO}]_{\mathcal{D},k} \subset [\text{SLR}]$, 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 gluing 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.

**[SO]_{\mathcal{D},k} \subset [\text{MSO}].** Since $[\text{SLR}] \subset [\text{SO}]$ and $[\text{SLR}]_{\mathcal{D},k} \subset [\text{MSO}]$, we obtain that $[\text{SO}]_{\mathcal{D},k} \subset [\text{MSO}]$. Moreover, $[\text{SO}] \subset [\text{MSO}]$ follows from the fact that our counterexample for $[\text{SLR}]_{\mathcal{D},k} \subset [\text{SO}]$ involves only structures of treewidth one.

**[SLR]_{\mathcal{D},k} \subset [\text{SO}].** By applying the translation of SLR to SO from §5 to $\Delta(k)$ (Fig. 1a) and to a given SID $\Delta$ 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 $[A()]_{\Delta}^{\mathcal{D},k}$, thus proving that $[\text{SLR}]_{\mathcal{D},k} \subset [\text{SO}]$.

**[(M)SO]_{\mathcal{D},k} \subset [(M)SO].** For each given $k \geq 1$, there exists an MSO formula $\theta_k$ 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 $\phi$, the (M)SO formula $\phi \land \theta_k$ defines the models of $\phi$ of treewidth at most $k$.

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

\(^4\) The proof of Robertson and Seymour does not build $\theta_k$, see [3] for an effective proof.
Open Problems. The following problems from Table 1 are currently open: $[\text{SLR}]^{D,k} \subseteq [\text{SLR}]$ and $[\text{SO}]^{D,k} \subseteq [\text{SLR}]$, both conjectured to have a negative answer. In particular, the difficulty concerning $[\text{SLR}]^{D,k} \subseteq [\text{SLR}]$ is that, in order to ensure treewidth boundedness, it seems necessary to force the composition of structures to behave like gluing (see the definition of $\Delta(k)$ in Fig. 1a), which seems difficult without the additional relation symbol $D$.

Since $[\text{MSO}]^{D,k} \subseteq [\text{SLR}]$ but $[\text{MSO}] \not\subseteq [\text{SLR}]$, 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

20:18 Expressiveness Results for an Inductive Logic of Separated Relations


