A Finitary Analogue of the Downward Löwenheim-Skolem Property

Abhisekh Sankaran

Department of Computer Science and Engineering, Indian Institute of Technology Bombay, Bombay, India
abhisekh@cse.iitb.ac.in

Abstract

We present a model-theoretic property of finite structures, that can be seen to be a finitary analogue of the well-studied downward Löwenheim-Skolem property from classical model theory. We call this property the *$\mathcal{L}$-equivalent bounded substructure property*, denoted $\mathcal{L}$-EBSP, where $\mathcal{L}$ is either FO or MSO. Intuitively, $\mathcal{L}$-EBSP states that a large finite structure contains a small "logically similar" substructure, where logical similarity means indistinguishability with respect to sentences of $\mathcal{L}$ having a given quantifier nesting depth. It turns out that this simply stated property is enjoyed by a variety of classes of interest in computer science: examples include regular languages of words, trees (unordered, ordered or ranked) and nested words, and various classes of graphs, such as cographs, graph classes of bounded tree-depth, those of bounded shrub-depth and n-partite cographs. Further, $\mathcal{L}$-EBSP remains preserved in the classes generated from the above by operations that are implementable using quantifier-free translation schemes. All of the aforementioned classes admit natural tree representations for their structures. We use this fact to show that the small and logically similar substructure of a large structure in any of these classes, can be computed in time linear in the size of the tree representation of the structure, giving linear time fixed parameter tractable (f.p.t.) algorithms for checking $\mathcal{L}$-definable properties of the large structure. We conclude by presenting a strengthening of $\mathcal{L}$-EBSP, that asserts "logical self-similarity at all scales" for a suitable notion of scale. We call this the *logical fractal* property and show that most of the classes mentioned above are indeed, logical fractals.

1998 ACM Subject Classification F.4.1 Model theory, F.4.3 Formal Languages, G.2.2 Graph theory

Keywords and phrases downward Löwenheim-Skolem theorem, trees, nested words, tree-depth, cographs, tree representation, translation schemes, composition lemma, f.p.t., logical fractal

Digital Object Identifier 10.4230/LIPIcs.CSL.2017.37

1 Introduction

The downward Löwenheim-Skolem theorem is one of the earliest results of classical model theory. This theorem, first proved by Löwenheim in 1915 [22], states that if a first order (henceforth, FO) theory over a countable vocabulary has an infinite model, then it has a countable model. In the mid-1920s, Skolem came up with a more general statement: any structure $\mathfrak{A}$ over a countable vocabulary has a countable "FO-similar" substructure. Here, "FO-similarity" of two given structures means that the structures agree on all properties that can be expressed in FO. This result of Skolem was further generalized by Mal'tsev in 1936 [24], to what is considered as the modern statement of the downward Löwenheim-Skolem theorem: for any infinite cardinal κ, any structure $\mathfrak{A}$ over a countable vocabulary has an elementary substructure (an FO-similar substructure having additional properties) that has size at most κ. The downward Löwenheim-Skolem theorem is one of the most important results of classical model theory, and indeed as Lindström showed in 1969 [21], FO is a maximal logic (having certain well-defined and reasonable closure properties) that satisfies this theorem, along with the (countable) compactness theorem.

The downward Löwenheim-Skolem theorem is intrinsically, a statement about infinite structures, and hence does not make sense in the finite when taken as is. While preservation and interpolation theorems from classical model theory have been actively studied over finite structures [16, 1, 26, 27, 3, 17], there is very little study of the downward Löwenheim-Skolem theorem (or adaptations of it) in the finite ([13, 34] seem to be the only studies of this theorem in the contexts of finite and pseudo-finite structures respectively). In this paper, we take a step towards addressing this issue. Specifically, we formulate a finitary analogue of the model-theoretic property contained in the downward Löwenheim-Skolem theorem, and show that classes of finite structures satisfying this analogue indeed abound in computer science. We call this analogue the *$\mathcal{L}$-equivalent bounded substructure property*, denoted $\mathcal{L}$-EBSP, where $\mathcal{L}$ is either FO or monadic second order logic (MSO). Intuitively, a class $\mathcal{S}$ of finite structures satisfies $\mathcal{L}$-EBSP if over $\mathcal{S}$, for each m, every structure $\mathfrak{A}$ contains a small substructure $\mathfrak{B}$ that is "$\mathcal{L}[m]$-similar" to $\mathfrak{A}$, where $\mathcal{L}[m]$ is the class of all sentences of $\mathcal{L}$ that have quantifier nesting depth at most m (Definition 3.1). In other words, $\mathfrak{B}$ and $\mathfrak{A}$ agree on all properties that can be described in $\mathcal{L}[m]$. The bound on the size of $\mathfrak{B}$ is given by a "witness function" that depends only on m (when $\mathcal{L}$ and $\mathcal{S}$ are fixed). It is easily seen that $\mathcal{L}$-EBSP has strong resemblance to the model-theoretic property contained in the downward Löwenheim-Skolem theorem, and can very well be seen as a finitary analogue of a version of this theorem that is "intermediate" between its versions by Skolem and Mal'tsev.

The motivation to define $\mathcal{L}$-EBSP came from our investigations over finite structures, of a generalization of the classical Łoś-Tarski preservation theorem from model theory, that was proved in [31]. This generalization, called the *generalized Łoś-Tarski theorem at level k*, denoted GLT(k), gives a semantic characterization, over arbitrary structures, of sentences in prenex normal form, whose quantifier prefixes are of the form $\exists^k\forall^*$, i.e. a sequence of k existential quantifiers followed by zero or more universal quantifiers. The Łoś-Tarski theorem is a special case of GLT(k) when k equals 0. Unfortunately, GLT(k) fails over all finite structures for all $k \geq 0$ (like most preservation theorems do [26]), and worse still, also fails for all $k \geq 2$, over the special classes of finite structures that are acyclic, of bounded degree, or of bounded tree-width, which were identified by Atserias, Dawar and Grohe [3] to satisfy the Łoś-Tarski theorem. This motivated the search for new (and possibly abstract) structural properties of classes of finite structures, that admit GLT(k) for each k. It is in this context that a version of $\mathcal{L}$-EBSP was first studied in [30]. The present paper takes that study much ahead. (Most of the results of this paper are from the author's Ph.D. thesis [28].)

Our results. We show that a variety of classes of finite structures of interest in computer science satisfy $\mathcal{L}$-EBSP, demonstrating that the latter property provides a unified framework, via logic, for studying these classes. The classes that we consider are broadly of two kinds: special kinds of labeled posets and special kinds of graphs. For the case of labeled posets, we show $\mathcal{L}$-EBSP holds for words, trees (of various kinds such as unordered, ordered, ranked, or "partially" ranked), and nested words over a finite alphabet, and all regular subclasses of these (Theorem 5.1). For each of these classes, we also show that $\mathcal{L}$-EBSP holds with computable witness functions. While words and trees have had a long history of studies in the literature, nested words are much recent [2], and have attracted a lot of attention as they admit a seamless generalization of the theory of regular languages, and are also closely connected with visibly pushdown languages. For the case of graphs, we show $\mathcal{L}$-EBSP holds for a very general, and again very recently defined, class of graphs called *n-partite cographs*, and all hereditary subclasses of this class (Theorem 5.2). This class of graphs, introduced in [12], jointly generalizes the classes of cographs (which includes several interesting graph classes such as complete r-partite graphs, Turan graphs, cluster graphs, threshold graphs, etc.), graph classes of bounded tree-depth and those of bounded shrub-depth. Cographs have been well studied since the '80s and have been shown to admit fast algorithms for many decision and optimization problems that are hard in general [19]. Graph classes of bounded tree-depth and bounded shrub-depth are much more recently defined [25, 12] and have become particularly prominent in the context of investigating fixed parameter tractable (f.p.t.) algorithms for MSO model checking, that have *elementary dependence* on the size of the MSO sentence (which is the parameter) [11, 12]. This line of work seeks to identify classes of structures for which Courcelle-style *algorithmic meta-theorems* [15] hold, but with better dependence on the parameter than in the case of Courcelle's theorem (which is unavoidably non-elementary [10]). A different and important line of work shows that FO and MSO are equal in their expressive powers over graph classes of bounded tree-depth/shrub-depth [11, 8]. Since each of the graph classes mentioned above is a hereditary subclass of the class of n-partite cographs for some n, each of these satisfies $\mathcal{L}$-EBSP, further with computable witness functions, and further still, with even elementary witness functions in many cases.

We give methods to construct new classes of structures satisfying $\mathcal{L}$-EBSP from those known to satisfy the latter property. Specifically, we show that $\mathcal{L}$-EBSP remains preserved under a wide range of operations on structures, that have been well-studied in the literature: unary operations like complementation, transpose and the line graph operation, binary "sum-like" operations [23] such as disjoint union and join, and binary "product-like" operations that include the Cartesian, tensor, lexicographic and strong products. All of these are examples of operations that can be implemented using *quantifier-free translation schemes* [23]. We show that FO-EBSP is always closed under such operations, and MSO-EBSP is closed under such operations, provided that they are unary or sum-like (Theorem 5.3). In both cases, the computability/elementariness of witness functions is preserved under the operations.

As algorithmic consequences of the above results, we obtain linear time f.p.t. algorithms for model checking $\mathcal{L}$-definable properties of structures, over all of the aforementioned classes. Each of these classes, including those generated using the various operations, admits natural tree representations for its structures. Specifically, for any structure, a tree representation of it is such that any leaf node of the tree is labeled with a substructure (typically a simple one), while any internal node is labeled with an operation that produces a new structure upon being fed as input, the structures represented by the children of the internal node. Our f.p.t. algorithms utilize the fact that the input structures are given in the form of these tree representations, to perform model checking in time linear in the sizes of the representations. The techniques used in our algorithms are based on the *composition method* from model theory [7, 23, 33, 15]. This method, made prominent by the work of Feferman and Vaught [9], allows inferring the sentences true in a structure composed of simpler structures, from the latter structures. In our context, the method allows determining the "$\mathcal{L}[m]$-similarity class" of the output structure of an operation, from the multi-set of the $\mathcal{L}[m]$-similarity classes of the structures that are input to the operation. For operations having arbitrary finite arity, the $\mathcal{L}[m]$-similarity class of the output is determined only by a threshold number of appearances of each $\mathcal{L}[m]$-similarity class in the multi-set, with the threshold depending solely on m. These technical features, in conjunction with the fact that index of the $\mathcal{L}[m]$-similarity relation is always finite, facilitate us in algorithmically *generating* the "composition functions" uniformly for any operation for any given m. The functions for any operation, in turn, enable doing the compositions in time linear in the arity of the operation. Using the latter fact, given a tree representation, we perform appropriate annotations, prunings and graftings in the tree iteratively, to produce in time linear in the size of the tree, a small subtree that represents a small $\mathcal{L}[m]$-similar substructure – the "kernel", in the f.p.t. parlance – of the structure represented by the given tree. Checking an $\mathcal{L}[m]$-definable property of the original structure then reduces to checking the same of the kernel. The above techniques have been incorporated into a single abstract result concerning tree representations (Theorem 4.2). Given that this result gives unified explanations for the good computational properties of many interesting classes, we believe it might be of independent interest.

Finally, we present a strengthened version of $\mathcal{L}$-EBSP, that turns out to be closely connected with the *fractal* property which has been extensively studied in mathematics, and in connection with a variety of natural phenomena [4]. Fractals are classes of mathematical structures that exhibit self-similarity at all scales. That is, every structure in the class contains a similar (in some technical sense) substructure at every scale of sizes less than the size of the structure. In this light, we observe that $\mathcal{L}$-EBSP indeed asserts "logical self-similarity" at "small scales". We formulate a strengthening of $\mathcal{L}$-EBSP, that asserts logical self-similarity at all scales, for a suitable notion of scale (Definition 6.1). We call this the *logical fractal* property, and call a class satisfying this property a *logical fractal.* It turns out that all of the aforementioned examples of poset and graph classes are logical fractals (Proposition 6.2). Further, all of the aforementioned operations on structures preserve the logical fractal property. The latter property thus appears to be naturally arising in diverse interesting settings of computer science. This suggests that adapting concepts from (mathematical) fractal theory (for instance, fractal dimension) to their logical counterparts can yield useful notions for finite model theory.

Paper organization. In Section 2, we introduce terminology and notation, and recall relevant notions from the literature used in the paper. In Section 3, we define $\mathcal{L}$-EBSP formally and show that it holds for the class of "partially" ranked trees, which are trees in which some subset of nodes are constrained to have degrees given by a ranking function. We use this special class as a setting to illustrate our techniques, that we lift to tree representations of structures in Section 4. In Section 5, we give applications of our abstract result to show $\mathcal{L}$-EBSP and the existence of linear time f.p.t. algorithms for $\mathcal{L}$-model checking, in various concrete settings, specifically those of posets and graphs mentioned earlier, and also classes constructed using various well-studied operations. We present the notion of logical fractals in Section 6, and conclude with open questions in Section 7.

2 Terminology and preliminaries

We assume familiarity with standard notions and notation of first order logic (FO) and monadic second order logic (MSO) [20]. By $\mathcal{L}$, we mean either FO or MSO. We consider only finite vocabularies, represented by τ or ν, that contain only *predicate symbols* (and no constant or function symbols), unless explicitly stated otherwise. All predicate symbols are assumed to have *positive* arity. We denote by $\mathcal{L}(\tau)$ the set of all $\mathcal{L}$ formulae over τ (and refer to these simply as $\mathcal{L}$ formulae, when τ is clear from context). A sequence $(x_1, \ldots, x_k)$ of variables is written as $\bar{x}$. A formula φ whose free variables are among $\bar{x}$, is denoted as $\varphi(\bar{x})$. Free variables are always *first order*. A formula with no free variables is called a *sentence*. The *rank* of an $\mathcal{L}$ formula is the maximum number of quantifiers (first order as well as second order) that appear along any path from the root to a leaf in the parse tree of the formula.

Finally, *a notion or result stated for $\mathcal{L}$ means that the notion or result respectively, is stated for both FO and MSO.*

Standard notions of τ-structures (denoted $\mathfrak{A}, \mathfrak{B}$ etc.; we refer to these simply as structures when τ is clear from context), substructures (denoted $\mathfrak{A} \subseteq \mathfrak{B}$) and extensions are used throughout the paper (see [20]). We assume all structures to be *finite*. As in [20], by substructures, we always mean *induced* substructures. Given a structure $\mathfrak{A}$, we use $\mathsf{U}_{\mathfrak{A}}$ to denote the universe of $\mathfrak{A}$, and $|\mathfrak{A}|$ to denote its cardinality. We denote by $\mathfrak{A} \cong \mathfrak{B}$ that $\mathfrak{A}$ is isomorphic to $\mathfrak{B}$, and by $\mathfrak{A} \hookrightarrow \mathfrak{B}$ that $\mathfrak{A}$ is isomorphically embeddable in $\mathfrak{B}$. For an $\mathcal{L}$ sentence φ, we denote by $\mathfrak{A} \models \varphi$ that $\mathfrak{A}$ is a model of φ. We denote classes of structures by $\mathcal{S}$, possibly with subscripts, and assume these to be *closed under isomorphisms.*

Let $\mathbb{N}$ and $\mathbb{N}_+$ denote the natural numbers including zero and excluding zero respectively. Given $m \in \mathbb{N}$ and a τ-structure $\mathfrak{A}$, denote by $\mathsf{Th}_{m,\mathcal{L}}(\mathfrak{A})$ the set of all $\mathcal{L}(\tau)$ sentences of rank at most m, that are true in $\mathfrak{A}$. Given a τ-structure $\mathfrak{B}$, we say that $\mathfrak{A}$ and $\mathfrak{B}$ are *$\mathcal{L}[m]$-equivalent*, denoted $\mathfrak{A} \equiv_{m,\mathcal{L}} \mathfrak{B}$ if $\mathsf{Th}_{m,\mathcal{L}}(\mathfrak{A}) = \mathsf{Th}_{m,\mathcal{L}}(\mathfrak{B})$. Given a class $\mathcal{S}$ of structures and $m \in \mathbb{N}$, we let $\Delta_{\mathcal{S},\mathcal{L},m}$ denote the set of all equivalence classes of the $\equiv_{m,\mathcal{L}}$ relation over $\mathcal{S}$. We denote by $\Lambda_{\mathcal{S},\mathcal{L}} : \mathbb{N} \rightarrow \mathbb{N}$ a fixed computable function with the property that $\Lambda_{\mathcal{S},\mathcal{L}}(m) \geq |\Delta_{\mathcal{S},\mathcal{L},m}|$. It is known that $\Lambda_{\mathcal{S},\mathcal{L}}$ always exists (see Proposition 7.5 in [20]). The notion of $\equiv_{m,\mathcal{L}}$ has a characterization using *Ehrenfeucht-Fräïssé* (EF) games for $\mathcal{L}$. We point the reader to Chapters 3 and 7 of [20] for results concerning these games.

We recall the notion of translation schemes from the literature [23] (known in the literature by different names, like *interpretations, transductions*, etc). Let τ and ν be given vocabularies, and $t \geq 1$ be a natural number. Let $\bar{x}_0$ be a fixed t-tuple of first order variables, and for each relation $R \in \nu$ of arity $\#R$, let $\bar{x}_R$ be a fixed $(t \times \#R)$-tuple of first order variables. A *$(t, \tau, \nu, \mathcal{L})$-translation scheme* $\Xi = (\xi, (\xi_R)_{R \in \nu})$ is a sequence of formulas of $\mathcal{L}(\tau)$ such that the free variables of ξ are among those in $\bar{x}_0$, and for $R \in \nu$, the free variables of ξ_R are among those in $\bar{x}_R$. When t, ν and τ are clear from context, we call Ξ simply a translation scheme, and t the *dimension* of Ξ. The translation scheme Ξ defines a map from τ-structures to ν-structures [23]. Abusing notation slightly, we denote this map as Ξ again, and for a class $\mathcal{S}$ of τ-structures, let $\Xi(\mathcal{S})$ denote the class $\{\Xi(\mathfrak{A}) \mid \mathfrak{A} \in \mathcal{S}\}$.

Given a class $\mathcal{S}$ of structures, we say that the model checking problem for $\mathcal{L}$ over $\mathcal{S}$, denoted $\mathsf{MC}(\mathcal{L}, \mathcal{S})$, is *fixed parameter tractable* [15], in short *f.p.t.*, if there exists an algorithm Alg that when given as input an $\mathcal{L}$ sentence φ of rank m, and a structure $\mathfrak{A} \in \mathcal{S}$, decides if $\mathfrak{A} \models \varphi$, in time $f(m) \cdot |\mathfrak{A}|^c$, where $f : \mathbb{N} \rightarrow \mathbb{N}$ is some computable function and c is a constant. In this case, we say Alg is an *f.p.t. algorithm* for $\mathsf{MC}(\mathcal{L}, \mathcal{S})$. We say Alg is a *linear time* f.p.t. algorithm for $\mathsf{MC}(\mathcal{L}, \mathcal{S})$ if it is f.p.t. for $\mathsf{MC}(\mathcal{L}, \mathcal{S})$ and runs in time $f(m) \cdot |\mathfrak{A}|$ where as before, f is a computable function and m is the rank of the input sentence.

The *k-fold* exponential function $\mathsf{exp}(n, k)$ is the function given inductively as: $\mathsf{exp}(n, 0) = n$ and $\mathsf{exp}(n, l) = 2^{\mathsf{exp}(n,l-1)}$ for $0 \leq l \leq k$. We call a function $f : \mathbb{N} \rightarrow \mathbb{N}$ *elementary* if there exists k such that $f(n) = O(\mathsf{exp}(n, k))$, and call it *non-elementary* if it is not elementary.

Finally, we use the following standard abbreviations throughout the paper: 'w.l.o.g' for 'without loss of generality', 'iff' for 'if and only if', and 'resp.' for 'respectively'.

3 The $\mathcal{L}$-Equivalent Bounded Substructure Property

▶ **Definition 3.1** ($\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$)**.** Let $\mathcal{S}$ be a class of structures and $\mathcal{L}$ be either FO or MSO. We say that $\mathcal{S}$ satisfies the *$\mathcal{L}$-equivalent bounded substructure property*, abbreviated $\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$ *is true* (alternatively, $\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$ *holds*), if there exists an increasing function $\theta_{(\mathcal{S},\mathcal{L})} : \mathbb{N} \rightarrow \mathbb{N}$ such that for each $m \in \mathbb{N}$ and each structure $\mathfrak{A}$ of $\mathcal{S}$, there exists a structure $\mathfrak{B}$ such that

(i) $\mathfrak{B} \in \mathcal{S}$, (ii) $\mathfrak{B} \subseteq \mathfrak{A}$, (iii) $|\mathfrak{B}| \leq \theta_{(\mathcal{S},\mathcal{L})}(m)$, and (iv) $\mathfrak{B} \equiv_{m,\mathcal{L}} \mathfrak{A}$. The conjunction of these four conditions is denoted as $\mathcal{L}$-EBSP-condition$(\mathcal{S}, \mathfrak{A}, \mathfrak{B}, m, \theta_{(\mathcal{S},\mathcal{L})})$. We call $\theta_{(\mathcal{S},\mathcal{L})}$ a *witness function for* $\mathcal{L}$-EBSP$(\mathcal{S})$, and say $\mathcal{L}$-EBSP$(\mathcal{S})$ *holds with a witness function* $\theta_{(\mathcal{S},\mathcal{L})}$.

We present below two simple examples of classes satisfying $\mathcal{L}$-EBSP.

1. Let $\mathcal{S}$ be the class of all τ-structures, where all predicates in τ are unary. By a simple FO-EF game argument, we see that FO-EBSP$(\mathcal{S})$ holds with a witness function $\theta_{(\mathcal{S},\mathrm{FO})} : \mathbb{N} \rightarrow \mathbb{N}$ given by $\theta_{(\mathcal{S},\mathrm{FO})}(m) = m \cdot 2^{|\tau|}$. In more detail: given $\mathfrak{A} \in \mathcal{S}$, associate exactly one of $2^{|\tau|}$ colors with each element a of $\mathfrak{A}$, where the colour gives the valuation of all predicates of τ for a in $\mathfrak{A}$. Then consider $\mathfrak{B} \subseteq \mathfrak{A}$ such that for each colour c, if $A_c = \{a \mid a \in \mathsf{U}_{\mathfrak{A}},\ a \text{ has colour } c \text{ in } \mathfrak{A}\}$, then $A_c \subseteq \mathsf{U}_{\mathfrak{B}}$ if $|A_c| < m$, else $|A_c \cap \mathsf{U}_{\mathfrak{B}}| = m$. It is easy to verify that FO-EBSP-condition$(\mathcal{S}, \mathfrak{A}, \mathfrak{B}, m, \theta_{(\mathcal{S},\mathrm{FO})})$ holds. By a similar MSO-EF game argument, MSO-EBSP$(\mathcal{S})$ holds with a witness function given by $\theta_{(\mathcal{S},\mathrm{MSO})}(m) = m \cdot 2^{(|\tau|+m)}$.
2. Let $\mathcal{S}$ be the class of disjoint unions of undirected paths. It can be easily shown that for any m, any two paths of length $\geq p = 3^m$ are FO$[m]$-equivalent. Let $\mathfrak{A} = \bigsqcup_{n\geq 0} i_n \cdot P_n$ where P_n denotes the path of length n, $i_n \cdot P_n$ denotes the disjoint union of i_n copies of P_n, and $\bigsqcup$ denotes disjoint union. For $n < p$, let j_n be such that $j_n = i_n$ if $i_n < m$ and $j_n = m$ if $i_n \geq m$. For $n = p$, let $j_n = h = \sum_{r \geq p} i_r$ if $h < m$, else $j_n = m$. If $\mathcal{B} = \bigsqcup_{n=1}^{n=p} j_n \cdot P_n$, then by an FO-EF game argument, $\mathfrak{B}$ satisfies FO-EBSP-condition$(\mathcal{S}, \mathfrak{A}, \mathfrak{B}, m, \theta_{(\mathcal{S},\mathrm{FO})})$ where $\theta_{(\mathcal{S},\mathrm{FO})}(m) = \sum_{n=0}^{n=p} m \cdot n$. Then FO-EBSP$(\mathcal{S})$ holds with a witness function $\theta_{(\mathcal{S},\mathrm{FO})}$.

3.1 Partially ranked trees satisfy $\mathcal{L}$-EBSP

In this subsection, we show that the class of ordered "partially" ranked trees satisfies $\mathcal{L}$-EBSP with computable witness functions, as well as admits a linear time f.p.t. algorithm for model checking $\mathcal{L}$ sentences. This setting illustrates our reasoning and techniques that we lift in Section 4 to the more abstract setting of tree representations of structures.

An *unlabeled unordered tree* is a finite poset $P = (A, \leq)$ with a unique minimal element (called "root"), such that for each $c \in A$, the set $\{b \mid b \leq c\}$ is totally ordered by $\leq$. Informally speaking, the Hasse diagram of P is an inverted (graph-theoretic) tree. We call A as the set of *nodes* of P. We use the standard notions of leaf, internal node, ancestor, descendent, parent, child, degree, height, and subtree in connection with trees. (We clarify that by height, we mean the maximum distance between the root and any leaf of the tree, as against the "number of levels" in the tree.) An *unlabeled ordered* tree is a pair $O = (P, \lesssim)$ where P is an unlabeled unordered tree and $\lesssim$ is a binary relation that imposes a linear order on the children of any internal node of P. Unless explicitly stated otherwise, we always consider our trees to be *ordered*. It is clear that the above mentioned notions in connection with unordered trees can be adapted for ordered trees. Given a countable alphabet Σ, a *tree over* Σ, also called a Σ*-tree*, or simply *tree* when Σ is clear from context, is a pair (O, λ) where O is an unlabeled tree and $\lambda : A \rightarrow \Sigma$ is a labeling function, where A is the set of nodes of O. We denote Σ-trees by $\mathsf{s}, \mathsf{t}, \mathsf{x}, \mathsf{y}, \mathsf{u}, \mathsf{v}$ or z, possibly with numbers as subscripts. Given a tree t, we denote the root of t as $\mathsf{root}(\mathsf{t})$. For a node a of t, we denote the subtree of t rooted at a as $\mathsf{t}_{\geq a}$, and the subtree of t obtained by deleting $\mathsf{t}_{\geq a}$ from t, as $\mathsf{t} - \mathsf{t}_{\geq a}$. Given a tree s and a non-root node a of t, the *replacement of* $\mathsf{t}_{\geq a}$ *with* s *in* t, denoted $\mathsf{t}\,[\mathsf{t}_{\geq a} \mapsto \mathsf{s}]$, is a tree defined as follows. Assume w.l.o.g. that s and t have disjoint sets of nodes. Let c be the parent of a in t. Then $\mathsf{t}\,[\mathsf{t}_{\geq a} \mapsto \mathsf{s}]$ is defined as the tree obtained by deleting $\mathsf{t}_{\geq a}$ from t to get a tree t', and inserting (the root of) s at the same position among the children of c in t', as the position of a among the children of c in t. For s and t as just mentioned, suppose the roots of both these trees have the same label. Then the *merge of* s *with* t, denoted $\mathsf{t} \odot \mathsf{s}$, is defined as the tree obtained by deleting $\mathsf{root}(\mathsf{s})$ from s and concatenating the sequence of subtrees hanging at $\mathsf{root}(\mathsf{s})$ in s, to the sequence of subtrees hanging at $\mathsf{root}(\mathsf{t})$ in t. Thus the children of $\mathsf{root}(\mathsf{s})$ in s are the "new" children of $\mathsf{root}(\mathsf{t})$, and appear "after" the "old" children of $\mathsf{root}(\mathsf{t})$, and in the order they appear in s.

Fix a finite alphabet Σ, and let $\Sigma_{\text{rank}} \subseteq \Sigma$. Let $\rho : \Sigma_{\text{rank}} \to \mathbb{N}_+$ be a fixed function. We say a Σ-tree $\mathsf{t} = (O, \lambda)$ is *partially ranked by* (Σ_{rank}, ρ) if for any node a of t, if $\lambda(a) \in \Sigma_{\text{rank}}$, then the number of children of a in t is exactly $\rho(\lambda(a))$. Observe that the case of $\Sigma_{\text{rank}} = \Sigma$ corresponds to the notion of ranked trees that are well-studied in the literature [5]. Let $\mathsf{Partially\text{-}ranked\text{-}trees}(\Sigma, \Sigma_{\text{rank}}, \rho)$ be the class of all ordered Σ-trees partially ranked by $(\Sigma_{\text{rank}}, \rho)$. The central result of this section is now as stated below.

▶ **Proposition 3.2.** *Given Σ, a subset Σ_{rank} of Σ and $\rho : \Sigma_{rank} \to \mathbb{N}_+$, let $\mathcal{S}$ be the class $\mathsf{Partially\text{-}ranked\text{-}trees}(\Sigma, \Sigma_{rank}, \rho)$. Then the following are true:*
1. *$\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$ holds with a computable witness function. Further, any witness function is necessarily non-elementary.*
2. *There is a linear time f.p.t. algorithm for $\mathsf{MC}(\mathcal{L}, \mathcal{S})$.*

We prove the two parts of the above result separately. In the remainder of this section, we fix $\mathcal{L}$, and also fix $\mathcal{S}$ to be the class $\mathsf{Partially\text{-}ranked\text{-}trees}(\Sigma, \Sigma_{\text{rank}}, \rho)$. Given these fixings, we denote $\Delta_{\mathcal{S},\mathcal{L},m}$ (the set of equivalence classes of the $\equiv_{m,\mathcal{L}}$ relation over $\mathcal{S}$) simply as Δ_m, and denote $\Lambda_{\mathcal{S},\mathcal{L}}(m)$ (see Section 2 for the definition of $\Lambda_{\mathcal{S},\mathcal{L}}(m)$) simply as $\Lambda(m)$.

▶ Remark. As mentioned in the introduction, our techniques to prove Proposition 3.2, are based on the "composition method" from model theory. One can also adopt automata based techniques to prove Proposition 3.2. Specifically, part (1) can be shown using the "downward direction" of the pumping lemma for such trees [5], while part (2) can be shown using the automata based approach described in [32]. Though, admittedly, the automata based approach is easier than the composition method for this special case of partially ranked trees, the machinery that we develop in this (illustrative) section admits a direct and seamless lifting to the abstract setting of tree representations that we consider in the next section. The abstract setting enables us to uniformly show our $\mathcal{L}$-EBSP and f.p.t. results for a wide range of interesting concrete instantiations that we present in Section 5.

Towards the proof of Proposition 3.2, we first present an $\mathcal{L}$-composition lemma for partially ranked trees, that is the "functional form" of a Feferman-Vaught (FV) style $\mathcal{L}$-composition lemma for these trees (See Appendix A for an FV-style $\mathcal{L}$-composition lemma for the more general case of ordered trees.) Recall that $\mathcal{S} = \mathsf{Partially\text{-}ranked\text{-}trees}(\Sigma, \Sigma_{\text{rank}}, \rho)$.

▶ **Lemma 3.3** (Composition lemma for partially ranked trees)**.** *For each $\sigma \in \Sigma$ and $m \geq 3$, there exists a function $f_{\sigma,m} : (\Delta_m)^{\rho(\sigma)} \to \Delta_m$ if $\sigma \in \Sigma_{rank}$, and functions $f_{\sigma,m,i} : (\Delta_m)^i \to \Delta_m$ for $i \in \{1, 2\}$ if $\sigma \in \Sigma \setminus \Sigma_{rank}$, with the following properties: Let $\mathsf{t} = (O, \lambda) \in \mathcal{S}$ and a be an internal node of t such that $\lambda(a) = \sigma$, and the children of a in t are $b_1, \ldots, b_n$. Let δ_i be the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{t}_{\geq b_i}$ for $i \in \{1, \ldots, n\}$, and let δ be the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{t}_{\geq a}$.*
1. *If $\sigma \in \Sigma_{rank}$ (whereby $n = \rho(\sigma)$), then $\delta = f_{\sigma,m}(\delta_1, \ldots, \delta_n)$.*
2. *If $\sigma \in \Sigma \setminus \Sigma_{rank}$, then δ is given as follows: For $k \in \{1, \ldots, n-1\}$, let $\chi_{k+1} = f_{\sigma,m,2}(\chi_k, \delta_{k+1})$ where $\chi_1 = f_{\sigma,m,1}(\delta_1)$. Then $\delta = \chi_n$.*

A useful corollary of this lemma is as below.

▶ **Corollary 3.4.** *The following are true for $m \geq 3$.*
1. *Given trees s, t and a non-root node a of t, let $\mathsf{z} = \mathsf{t}[\mathsf{t}_{\geq a} \mapsto \mathsf{s}]$. If $\mathsf{s} \equiv_{m,\mathcal{L}} \mathsf{t}_{\geq a}$, then $\mathsf{z} \equiv_{m,\mathcal{L}} \mathsf{t}$.*

2. *Let* $\mathsf{s}_1, \mathsf{s}_2, \mathsf{t}$ *be given trees such that the labels of their roots are the same, and belong to* $\Sigma \setminus \Sigma_{rank}$. *Suppose* $\mathsf{z}_i = \mathsf{s}_i \odot \mathsf{t}$ *for* $i \in \{1,2\}$. *If* $\mathsf{s}_1 \equiv_{m,\mathcal{L}} \mathsf{s}_2$, *then* $\mathsf{z}_1 \equiv_{m,\mathcal{L}} \mathsf{z}_2$.
3. *Let* $\mathsf{s}_1, \mathsf{s}_2$ *be given trees such that the labels of their roots are the same, and belong to* $\Sigma \setminus \Sigma_{rank}$. *For* $i \in \{1,2\}$, *given* t_i, *let* z_i *be the tree obtained from* s_i *by adding* t_i *as the (new) "last" child subtree of the root of* s_i. *If* $\mathsf{s}_1 \equiv_{m,\mathcal{L}} \mathsf{s}_2$ *and* $\mathsf{t}_1 \equiv_{m,\mathcal{L}} \mathsf{t}_2$, *then* $\mathsf{z}_1 \equiv_{m,\mathcal{L}} \mathsf{z}_2$.

Proof sketch for part (1) of Proposition 3.2: The first half follows from the following "reduction" lemma for trees. The second half follows from the fact that even over Σ-words (Σ-labeled linear orders), the index of the $\equiv_{m,\mathcal{L}}$ relation depends non-elementarily on m [10].

▶ **Lemma 3.5.** *There exist computable functions* $\eta_1, \eta_2 : \mathbb{N} \to \mathbb{N}$ *such that for each* $\mathsf{t} \in \mathcal{S}$ *and* $m \in \mathbb{N}$, *the following hold:*
1. (Degree reduction) *There exists a subtree* s_1 *of* t *in* $\mathcal{S}$, *of degree* $\leq \eta_1(m)$, *such that* (i) *the roots of* s_1 *and* t *are the same, and* (ii) $\mathsf{s}_1 \equiv_{m,\mathcal{L}} \mathsf{t}$.
2. (Height reduction) *There exists a subtree* s_2 *of* t *in* $\mathcal{S}$, *of height* $\leq \eta_2(m)$, *such that* (i) *the roots of* s_2 *and* t *are the same, and* (ii) $\mathsf{s}_2 \equiv_{m,\mathcal{L}} \mathsf{t}$.

Proof. For a finite subset X of $\mathbb{N}$, let $\max(X)$ denote the maximum element of X.

(1) For $n \geq 3$, define $\eta_1(n) = \max(\{\rho(\sigma) \mid \sigma \in \Sigma_{\text{rank}}\} \cup \{3\}) \times \Lambda(n)$. For $n < 3$, define $\eta_1(n) = \eta_1(3)$. We prove this part for $m \geq 3$; then it follows that this part is also true for $m < 3$ (by taking s_1 for the $m = 3$ case as s_1 for the $m < 3$ case).
Given $m \geq 3$, let $p = \eta_1(m)$. If t has degree $\leq p$, then putting $\mathsf{s}_1 = \mathsf{t}$, we are done. Else, some node a of t has degree $n > p$. Clearly then $\lambda(a) \notin \Sigma_{\text{rank}}$. Let $\mathsf{z} = \mathsf{t}_{\geq a}$ and let $a_1, \ldots, a_n$ be the (ascending) sequence of children of $\mathsf{root}(\mathsf{z})$ in z. For $1 \leq j \leq n$, let $\mathsf{x}_{1,j}$, resp. $\mathsf{y}_{j+1,n}$, be the subtree of z obtained from z by deleting the subtrees rooted at $a_{j+1}, \ldots, a_n$, resp. deleting the subtrees rooted at $a_1, a_2, \ldots, a_j$. Then $\mathsf{z} = \mathsf{x}_{1,n} = \mathsf{x}_{1,j} \odot \mathsf{y}_{j+1,n}$ for $1 \leq j < n$. Let $g : \{1, \ldots, n\} \to \Delta_m$ be such that $g(j)$ is the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{x}_{1,j}$. Since $n > p$, there exist $j, k \in \{1, \ldots, n\}$ such that $j < k$ and $g(j) = g(k)$, i.e. $\mathsf{x}_{1,j} \equiv_{m,\mathcal{L}} \mathsf{x}_{1,k}$. If $k < n$, then let $\mathsf{z}_1 = \mathsf{x}_{1,j} \odot \mathsf{y}_{k+1,n}$, else let $\mathsf{z}_1 = \mathsf{x}_{1,j}$. Then by Corollary 3.4, $\mathsf{z}_1 \equiv_{m,\mathcal{L}} \mathsf{z}$. Let t_1 be the subtree of t in $\mathcal{S}$ given by $\mathsf{t}_1 = \mathsf{t}\,[\mathsf{z} \mapsto \mathsf{z}_1]$. By Corollary 3.4 again, $\mathsf{t}_1 \equiv_{m,\mathcal{L}} \mathsf{t}$. Observe that t_1 has strictly lesser size than t. Recursing on t_1, we are eventually done.

(2) For $n \geq 3$, define $\eta_2(n) = \Lambda(n) + 1$. For $n < 3$, define $\eta_2(n) = \eta_2(3)$. As before, it suffices to prove this part for $m \geq 3$.
Given $m \geq 3$, let $p = \eta_2(m)$. If t has height $\leq p$, then putting $\mathsf{s}_2 = \mathsf{t}$, we are done. Else, there is a path from the root of t to some leaf of t, whose length is $> p$. Let A be the set of nodes appearing along this path. Let $h : A \to \Delta_m$ be such that for each $a \in A$, $h(a)$ is the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{t}_{\geq a}$. Since $|A| > p$, there exist distinct nodes $a, b \in A$ such that a is an ancestor of b in t, $a \neq \mathsf{root}(\mathsf{t})$, and $h(a) = h(b)$. Let $\mathsf{t}_2 = \mathsf{t}\,[\mathsf{t}_{\geq a} \mapsto \mathsf{t}_{\geq b}]$; then t_2 is a subtree of t in $\mathcal{S}$. Since $h(a) = h(b)$, $\mathsf{t}_{\geq a} \equiv_{m,\mathcal{L}} \mathsf{t}_{\geq b}$. By Corollary 3.4, we get $\mathsf{t}_2 \equiv_{m,\mathcal{L}} \mathsf{t}$. Note that t_2 has strictly lesser size than t. Recursing on t_2, we are eventually done.

◀

Proof sketch for part (2) of Proposition 3.2: The following result contains the core argument for the proof of this part. The first part of Lemma 3.6 gives an algorithm to generate the "composition" functions of Lemma 3.3, uniformly for $m \geq 3$. This algorithm is in turn used in the second part of Lemma 3.6 to get a "linear time" version of Lemma 3.5.

▶ **Lemma 3.6.** *There exist computable functions* $\eta_3, \eta_4, \eta_5 : \mathbb{N} \to \mathbb{N}$ *and algorithms* Generate-functions(m), Reduce-degree(t, m) *and* Reduce-height(t, m) *such that for* $m \geq 3$,

1. $\mathsf{Generate\text{-}functions}(m)$ *generates in time* $\eta_3(m)$*, the functions* $f_{\sigma,m}$ *if* $\sigma \in \Sigma_{rank}$ *and* $f_{\sigma,m,i}$ *for* $i \in \{1, 2\}$ *if* $\sigma \in \Sigma \setminus \Sigma_{rank}$*, that satisfy the properties mentioned in Lemma 3.3.*
2. *For* $\mathsf{t} \in \mathcal{S}$*,* $\mathsf{Reduce\text{-}degree}(\mathsf{t}, m)$ *computes the subtree* s_1 *of* t *as given by Lemma 3.5, in time* $\eta_4(m) \cdot |\mathsf{t}|$*. Likewise,* $\mathsf{Reduce\text{-}height}(\mathsf{t}, m)$ *computes the subtree* s_2 *of* t *as given by Lemma 3.5, in time* $\eta_5(m) \cdot |\mathsf{t}|$*.*

Proof Sketch.

(1) We first observe that the $\mathcal{L}$-SAT problem is decidable over $\mathcal{S}$ – since $\mathcal{L}$-EBSP($\mathcal{S}$) holds with a computable witness function (by Proposition 3.2(1)), if an $\mathcal{L}$ sentence has a model in $\mathcal{S}$, it also has a model of size bounded by a computable function of its rank.

Generate-functions(m)

1. Create a list $\mathcal{L}$[m]-classes of the $\equiv_{m,\mathcal{L}}$ classes over $\mathcal{S}$. This is done as follows:
   a. Given the inductive definition of $\mathcal{L}[m]$, there is an algorithm $\mathcal{P}(m)$ that enumerates $\mathcal{L}[m]$ sentences $\varphi_1, \varphi_2, \ldots, \varphi_n$ such that each φ_i captures some class in $\Delta_{\mathsf{All},\mathcal{L},m}$ (the set of equivalence classes of the $\equiv_{m,\mathcal{L}}$ relation over all finite structures), and conversely, each class of $\Delta_{\mathsf{All},\mathcal{L},m}$ is captured by some φ_i. First invoke $\mathcal{P}(m)$ to get the φ_is.
   b. For each $i \in \{1, \ldots, n\}$, if φ_i is satisfiable over $\mathcal{S}$ (whereby it represents some equivalence class of the $\equiv_{m,\mathcal{L}}$ relation over $\mathcal{S}$), then put it in $\mathcal{L}$[m]-classes, else discard it. (We interchangeably regard $\mathcal{L}$[m]-classes as a list of $\mathcal{L}[m]$ sentences or a list of $\equiv_{m,\mathcal{L}}$ classes.)
2. For $\sigma \in \Sigma_{\text{rank}}$ and $d = \rho(\sigma)$, generate $g_{\sigma,m} : (\mathcal{L}\text{[m]-classes})^d \to \mathcal{L}\text{[m]-classes}$ as follows. Given $\xi_i \in \mathcal{L}$[m]-classes for $i \in \{1, \ldots, d\}$, find models s_i for ξ_i in $\mathcal{S}$. Let s be the tree obtained by making $\mathsf{s}_1, \ldots, \mathsf{s}_n$ as the child subtrees (and in that sequence) of a new root node labeled with σ. Find out $\xi \in \mathcal{L}$[m]-classes of which s is a model. Then define $g_{\sigma,m}(\xi_1, \ldots, \xi_d) = \xi$. Generate $g_{\sigma,m,1} : \mathcal{L}\text{[m]-classes} \to \mathcal{L}\text{[m]-classes}$ similarly.
3. For $\sigma \in \Sigma \setminus \Sigma_{\text{rank}}$, generate $g_{\sigma,m,2} : (\mathcal{L}\text{[m]-classes})^2 \to \mathcal{L}\text{[m]-classes}$ as follows. For $\xi_1, \xi_2 \in \mathcal{L}$[m]-classes, find models s_1 and s_2 resp. in $\mathcal{S}$. such that the root of s_1 is labeled with σ (this condition on the root can be captured by an FO sentence). If no s_1 is found, then define $g_{\sigma,m,2}(\xi_1, \xi_2) = \xi_{\text{default}}$ where the latter is some fixed element of $\mathcal{L}$[m]-classes. Else, let v_{ξ_1,ξ_2} be the tree obtained adding s_2 as the (new) "last" child subtree of the root of s_1. Find out $\xi \in \mathcal{L}$[m]-classes of which v_{ξ_1,ξ_2} is a model. Define $g_{\sigma,m,2}(\xi_1, \xi_2) = \xi$.

It is clear that there exists a computable function $\eta_3 : \mathbb{N} \to \mathbb{N}$ such that the running time of Generate-functions(m) is at most $\eta_3(m)$. We now claim that $g_{\sigma,m}$ and $g_{\sigma,m,i}$ generated by Generate-functions(m) indeed satisfy the composition properties of Lemma 3.3, whereby they can be indeed taken as $f_{\sigma,m}$ and $f_{\sigma,m,i}$ appearing in the latter lemma. That $g_{\sigma,m}$ and $g_{\sigma,m,1}$ satisfy the composition properties is easy to see using Corollary 3.4. To reason for $g_{\sigma,m,2}$, consider a tree t whose root is labeled with σ, and which has say 3 children $a_1, \ldots, a_3$ (and in that sequence) such that the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{t}_{\geq a_i}$ is δ_i for $1 \leq i \leq 3$. Consider the subtrees x and y of t defined as $\mathsf{x} = \mathsf{t} - \mathsf{t}_{\geq a_3}$ and $\mathsf{y} = \mathsf{x} - \mathsf{x}_{\geq a_2}$. Let δ_4 and δ_5 be resp. the $\equiv_{m,\mathcal{L}}$ classes of x and y. Now consider the trees $\mathsf{v}_{\delta_5,\delta_2}$ and $\mathsf{v}_{\delta_4,\delta_3}$ which are guaranteed to be found (since indeed x and y *are* trees each of whose roots is labeled with σ). By Corollary 3.4, $\mathsf{x} \equiv_{m,\mathcal{L}} \mathsf{v}_{\delta_5,\delta_2}$ and $\mathsf{t} \equiv_{m,\mathcal{L}} \mathsf{v}_{\delta_4,\delta_3}$. Whereby, the $\equiv_{m,\mathcal{L}}$ class of x is $\delta_4 = g_{\sigma,m,2}(\delta_5, \delta_2)$ and that of t is $\delta = g_{\sigma,m,2}(\delta_4, \delta_3)$. Observe that δ_5 is indeed $g_{\sigma,m,1}(\delta_1)$.

(2) Reduce-degree(t, m)

```
1. Call Generate-functions($m$) that returns the "composition" functions $f_{\sigma,m}$ and $f_{\sigma,m,i}$,
   and also gives the list $\mathcal{L}$[m]-classes as described above.
2. Using the composition functions, construct bottom-up in t, the function Colour :
   Nodes(t) $\rightarrow \mathcal{L}$[m]-classes such that for each node $a$ of t, Colour($a$) is the $\equiv_{m,\mathcal{L}}$ class
   of $\mathsf{t}_{\geq a}$.
3. For $\eta_1$ as given by Lemma 3.5, if the degree of t is $\leq \eta_1(m)$, then return t.
4. Else, let $a$ be a node of t of degree $n > \eta_1(m)$. Let $\mathsf{x} = \mathsf{t}_{\geq a}$.
5. For each $\delta \in \mathcal{L}$[m]-classes, do the following:
   a. Let $a_1, \ldots, a_n$ be the children of $a$ in x. For $k \in \{1, \ldots, n\}$, let $\mathsf{x}_{1,k}$ be the
      subtree of x obtained by deleting the subtrees rooted at $a_{k+1}, \ldots, a_n$. Let
      $g : \{1, \ldots, n\} \rightarrow \mathcal{L}$[m]-classes be such that $g(i)$ is the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{x}_{1,k}$.
   b. If $\delta$ appears in the range of $g$, then let $i, j$ be resp. the least and greatest indices
      in $\{1, \ldots, n\}$ such that $g(i) = g(j) = \delta$. Let y be the subtree of x obtained by
      deleting the subtrees rooted at $a_{i+1}, \ldots, a_j$. Set x := y.
6. Set $\mathsf{t} := \mathsf{t}[\mathsf{t}_{\geq a} \mapsto \mathsf{x}]$ and go to step 3.
```

Reasoning similarly as for Lemma 3.5(1), we can verify that Reduce-degree(t, m) indeed returns the desired subtree s_1 of t. The time taken to compute Colour is linear in $|\mathsf{t}|$, while that for computing g is linear in the degree of a, whereby the time taken to reduce the degree of a node a in any iteration of the loop, is $O(\Lambda(m) \cdot \text{degree}(a))$. Then, the total time taken by Reduce-degree(t, m) is $O(\alpha(m) + \Lambda(m) \cdot |\mathsf{t}|)$ for some computable function $\alpha : \mathbb{N} \rightarrow \mathbb{N}$.

Reduce-height(t, m)

```
1. Generate $\mathcal{L}$[m]-classes and the function Colour as in the previous part.
2. Construct bottom up in t, the function Lowest-subtree : Nodes(t) $\times \mathcal{L}$[m]-classes $\rightarrow$
   Nodes(t) such that for any node $a$ of t and $\delta \in \mathcal{L}$[m]-classes, Lowest-subtree($a, \delta$)
   gives a lowest (i.e. closest to a leaf) node $b$ in $\mathsf{t}_{\geq a}$ such that Colour($b$) = $\delta$. In other
   words, $b$ is the only node in $\mathsf{t}_{\geq b}$ such that Colour($b$) = $\delta$.
3. Let $a_1, \ldots, a_n$ be the children of root(t). Let $\mathsf{x}_i$ = Rainbow-subtree($\mathsf{t}_{\geq a_i}$) for $i \in$
   $\{1, \ldots, n\}$, where Rainbow-subtree(x) is described below.
4. Return $\mathsf{t}[\mathsf{t}_{\geq a_1} \mapsto \mathsf{x}_1] \ldots [\mathsf{t}_{\geq a_n} \mapsto \mathsf{x}_n]$.
```

Rainbow-subtree(x)

```
1. Let $a$ = root(x).
2. If $b$ = Lowest-subtree($a$, Colour($a$)) $\neq a$, then return Rainbow-subtree($\mathsf{x}_{\geq b}$).
3. Else, let $b_1, \ldots, b_n$ be the children of root(x). For $1 \leq i \leq n$, let
   $\mathsf{y}_i$ = Rainbow-subtree($\mathsf{x}_{\geq b_i}$).
4. Return $\mathsf{x}[\mathsf{x}_{\geq b_1} \mapsto \mathsf{y}_1] \ldots [\mathsf{x}_{\geq b_n} \mapsto \mathsf{y}_n]$.
```

Using similar reasoning as in the proof of Lemma 3.5(2), we can verify that algorithm Rainbow-subtree(x), that takes a subtree x of t as input, outputs a subtree y of x such that (i) $\mathsf{y} \equiv_{m,\mathcal{L}} \mathsf{x}$ and (ii) no path from the root to the leaf of y contains two distinct nodes a and b such that Colour(a) = Colour(b). Further, Rainbow-subtree(x) also satisfies the following "colour preservation" property. Let for a subtree s of t, obtained from t by removal of rooted subtrees and replacements with rooted subtrees, $\mathcal{Q}$(s) be a predicate that denotes that the function Colour computed for t, when restricted to the nodes of s, is such that for any node a of s, Colour(a) gives the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{s}_{\geq a}$. Then the "colour preservation" property says that if the input x to $\mathsf{Rainbow\text{-}subtree}$ satisfies $\mathcal{Q}(\cdot)$, then so does the output y of $\mathsf{Rainbow\text{-}subtree}$.

From the preceding features of $\mathsf{Rainbow\text{-}subtree}$, we see that the height of the output y of $\mathsf{Rainbow\text{-}subtree}(\mathsf{x})$ is at most $\Lambda(m)$. The number of "top level" recursive calls made by $\mathsf{Rainbow\text{-}subtree}(\mathsf{x})$ is linear in the degree of $\mathsf{root}(\mathsf{x})$, whereby the total time taken by $\mathsf{Rainbow\text{-}subtree}(\mathsf{x})$ is linear in $|\mathsf{x}|$. The time taken to compute $\mathsf{Lowest\text{-}subtree}$ is easily seen to be $O(\Lambda(m) \cdot |\mathsf{t}|)$. Then the time taken by $\mathsf{Reduce\text{-}height}(\mathsf{t}, m)$ is $O(\eta_3(m) + \Lambda(m) \cdot |\mathsf{t}|)$. One can verify that $\mathsf{Reduce\text{-}height}(\mathsf{t}, m)$ indeed returns the desired subtree s_2 of t. ◀

4 Lifting to tree representations

We now consider the more abstract setting of tree representations of structures, and show that under suitable conditions on these representations (that a variety of classes of structures satisfy), we can lift the techniques seen in the previous section. Fix finite alphabets Σ_{int} and Σ_{leaf} (where the two alphabets are allowed to be overlapping). Let $\Sigma_{\text{rank}} \subseteq \Sigma_{\text{int}}$. Let $\rho : \Sigma_{\text{int}} \to \mathbb{N}_+$ be a fixed function. We say a class $\mathcal{T}$ of $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$-trees is *representation-feasible for* (Σ_{rank}, ρ) if $\mathcal{T}$ is closed under (label-preserving) isomorphisms, and for all trees $\mathsf{t} = (O, \lambda) \in \mathcal{T}$ and nodes a of t, the following conditions hold:

1. Labeling condition: If a is a leaf node, resp. internal node, then the label $\lambda(a)$ belongs to Σ_{leaf}, resp. Σ_{int}.
2. Ranking by ρ: If a is an internal node and $\lambda(a)$ is in Σ_{rank}, then the number of children of a in t is exactly $\rho(\lambda(a))$.
3. Closure under rooted subtrees: The subtree $\mathsf{t}_{\geq a}$ is in $\mathcal{T}$.
4. Closure under removal of rooted subtrees respecting Σ_{rank}: If a is an internal node, b is a child of a in t and $\lambda(a) \notin \Sigma_{\text{rank}}$, then the subtree $(\mathsf{t} - \mathsf{t}_{\geq b})$ is in $\mathcal{T}$.
5. Closure under replacements with rooted subtrees: If a is an internal node, then for every descendent b of a in t, the subtree $\mathsf{t}\,[\mathsf{t}_{\geq a} \mapsto \mathsf{t}_{\geq b}]$ is in $\mathcal{T}$.

Given such a class $\mathcal{T}$ of trees as above and a class $\mathcal{S}$ of structures, let $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$ be a map that associates with each tree in $\mathcal{T}$, a structure in $\mathcal{S}$. We call Str a *representation map*. For a tree $\mathsf{t} \in \mathcal{T}$, if $\mathfrak{A} = \mathsf{Str}(\mathsf{t})$, then we say t is a *tree representation* of $\mathfrak{A}$ under Str. For the purposes of our result, we consider "good" maps that would allow tree reductions of the kind seen in the previous section. We formally define these below:

▶ **Definition 4.1** ($\mathcal{L}$-good representation maps). Let $\mathcal{S}$ be a class of structures and $\mathcal{T}$ be a class of $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$-trees that is representation-feasible for $(\Sigma_{\text{rank}}, \rho)$. A representation map $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$ is said to be $\mathcal{L}$-*good* if it has the following properties:

1. Isomorphism preservation: Str maps isomorphic (labeled) trees to isomorphic structures.
2. Surjectivity: Each structure in $\mathcal{S}$ has an isomorphic structure in the range of Str.
3. Monotonicity: Let $\mathsf{t} \in \mathcal{T}$ be a tree of size ≥ 2, and a be a node of t.
   a. If $\mathsf{s} = \mathsf{t}_{\geq a}$, then $\mathsf{Str}(\mathsf{s}) \hookrightarrow \mathsf{Str}(\mathsf{t})$.
   b. If b is a child of a in t, $\lambda(a) \notin \Sigma_{\text{rank}}$ and $\mathsf{z} = (\mathsf{t} - \mathsf{t}_{\geq b})$, then $\mathsf{Str}(\mathsf{z}) \hookrightarrow \mathsf{Str}(\mathsf{t})$.
   c. If b is a descendent of a in t and $\mathsf{z} = \mathsf{t}\,[\mathsf{t}_{\geq a} \mapsto \mathsf{t}_{\geq b}]$, then $\mathsf{Str}(\mathsf{z}) \hookrightarrow \mathsf{Str}(\mathsf{t})$.
4. Composition: There exists $m_0 \in \mathbb{N}$ such that for every $m \geq m_0$ and for every $\sigma \in \Sigma_{\text{int}}$, there exists a function $f_{\sigma,m} : (\Delta_{\mathcal{S},\mathcal{L},m})^{\rho(\sigma)} \to \Delta_{\mathcal{S},\mathcal{L},m}$ if $\sigma \in \Sigma_{\text{rank}}$, and functions $f_{\sigma,m,i} : (\Delta_{\mathcal{S},\mathcal{L},m})^{i} \to \Delta_{\mathcal{S},\mathcal{L},m}$ for $i \in \{1, \ldots, \rho(\sigma)\}$ if $\sigma \in \Sigma_{\text{int}} \setminus \Sigma_{\text{rank}}$, with the following properties: Let $\mathsf{t} = (O, \lambda) \in \mathcal{T}$ and a be an internal node of t such that $\lambda(a) = \sigma$, and let the children of a in t be $b_1, \ldots, b_n$. Let δ_i be the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{Str}(\mathsf{t}_{\geq b_i})$ for $i \in \{1, \ldots, n\}$, and let δ be the $\equiv_{m,\mathcal{L}}$ class of $\mathsf{Str}(\mathsf{t}_{\geq a})$.

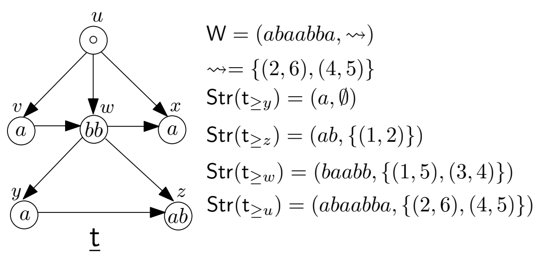

Figure 1 Nested word as a tree.

- If $\sigma \in \Sigma_{\text{rank}}$ (whereby $n = \rho(\sigma)$), then $\delta = f_{\sigma,m}(\delta_1, \ldots, \delta_n)$.
- If $\sigma \in \Sigma_{\text{int}} \setminus \Sigma_{\text{rank}}$, then δ is given as follows: Let $d = \rho(\sigma)$ and $n = r + q \cdot (d-1)$ where $1 \leq r < d$. Let $I = \{r + j \cdot (d-1) \mid 0 \leq j \leq q\}$ and for $k \in I, k \neq n$, let $\chi_{k+(d-1)} = f_{\sigma,m,d}(\chi_k, \delta_{k+1}, \ldots, \delta_{k+(d-1)})$ where $\chi_r = f_{\sigma,m,r}(\delta_1, \ldots, \delta_r)$. Then $\delta = \chi_n$.

We say $\mathcal{S}$ *admits an $\mathcal{L}$-good tree representation* if there exists a representation map $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$ that is $\mathcal{L}$-good. We say an $\mathcal{L}$-good representation map $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$ is *effective (resp. elementary)* if (i) $\mathcal{T}$ is recursive and (ii) there is an algorithm that, given $\mathsf{t} \in \mathcal{T}$ as input, computes $\mathsf{Str}(\mathsf{t})$ (resp. computes $\mathsf{Str}(\mathsf{t})$ in time that is bounded by an elementary function of $|\mathsf{t}|$). We now present the central result of this section, which is a lifting of Proposition 3.2 to tree representations. The proof involves an abstraction of all the ideas presented in proof of Proposition 3.2. The details of the proof can be found in [29].

▶ **Theorem 4.2.** *Let $\mathcal{S}$ be a class of structures that admits an $\mathcal{L}$-good tree representation $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$. Then the following are true:*
1. *$\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$ holds.*
2. *If Str is effective, then there exists a computable witness function for $\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$. Further, there exists a linear time f.p.t. algorithm for $\mathsf{MC}(\mathcal{L}, \mathcal{S})$ that decides, for every $\mathcal{L}$ sentence φ (the parameter), if a given structure $\mathfrak{A}$ in $\mathcal{S}$ satisfies φ, provided that $\mathfrak{A}$ is given in the form of a tree representation of it under Str.*
3. *If Str is elementary, then there exists an elementary witness function for $\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$ iff the index of the $\equiv_{m,\mathcal{L}}$ relation over $\mathcal{S}$ has an elementary dependence on m.*

5 Applications to various concrete settings

A. Regular languages of words, trees and nested words. Let Σ be a finite alphabet. The notion of unordered, ordered, ranked and partially ranked Σ-trees was already introduced in Section 3.1. A Σ-tree whose underlying poset is a linear order is called a Σ-*word*. A *nested word over* Σ is a pair $(w, \rightsquigarrow)$ where w is a "usual" Σ-word and $\rightsquigarrow$ is a binary relation representing a "nested matching". Formally, if $(A, \leq)$ is the linear order underlying w, then $\rightsquigarrow$ satisfies the following: (i) for $i, j \in A$, if $i \rightsquigarrow j$, then $i \leq j$ and $i \neq j$ (ii) for $i \in A$, there is at most one $j \in A$ such that $i \rightsquigarrow j$ and at most one $l \in A$ such that $l \rightsquigarrow i$, and (iii) for $i_1, i_2, j_1, j_2 \in A$, if $i_1 \rightsquigarrow j_1$ and $i_2 \rightsquigarrow j_2$, then it is not the case that $i_1 < i_2 \leq j_1 < j_2$. (Nested words here correspond to those of [2] that have no pending calls or pending returns.)

For e.g., $\mathsf{w} = (abaabba, \{(2,6),(4,5)\})$ is a nested word over $\{a, b\}$. A nested Σ-word has a natural representation using a tree over $\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}}$, where $\Sigma_{\text{leaf}} = \Sigma \cup (\Sigma \times \Sigma)$, and $\Sigma_{\text{int}} = (\Sigma \times \Sigma) \cup \{\circ\}$. Figure 1 shows such a tree t for w. Conversely, every ordered tree over $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$, whose leaf and internal node labels belong to Σ_{leaf} and Σ_{int} resp., represents a nested Σ-word.

The notion of *regular languages* of words, trees (of all the aforementioned kinds) and nested words, and its correspondence with MSO definability, are well-studied [5, 2]. We say a class of words, trees or nested words is regular if it is the class of models of an MSO sentence.

▶ **Theorem 5.1.** *Given finite alphabets Σ, Ω such that $\Omega \subseteq \Sigma$, and a function $\rho : \Omega \to \mathbb{N}$, let $\mathsf{Words}(\Sigma)$, $\mathsf{Unordered\text{-}trees}(\Sigma)$, $\mathsf{Ordered\text{-}trees}(\Sigma)$, $\mathsf{Partially\text{-}ranked\text{-}trees}(\Sigma, \Omega, \rho)$ and $\mathsf{Nested\text{-}words}(\Sigma)$ denote resp. the classes of all Σ-words, all unordered Σ-trees, all ordered Σ-trees, all ordered Σ-trees partially ranked by (Ω, ρ), and all nested Σ-words. Let $\mathcal{S}$ be a regular subclass of any of these classes. Then $\mathcal{L}\text{-}\mathsf{EBSP}(\mathcal{S})$ holds with a computable witness function. Further, any witness function for $\mathcal{L}\text{-}\mathsf{EBSP}(\mathcal{S})$ is necessarily non-elementary.*

Proof Idea. We first show MSO-$\mathsf{EBSP}(\mathcal{S})$ holds when $\mathcal{S}$ is exactly one of the classes mentioned in the statement of the theorem. That $\mathcal{L}\text{-}\mathsf{EBSP}(\cdot)$ holds for a regular subclass follows, because (i) MSO-$\mathsf{EBSP}(\cdot)$ is preserved under MSO definable subclasses, and (ii) MSO-$\mathsf{EBSP}(\cdot)$ implies FO-$\mathsf{EBSP}(\cdot)$. Consider $\mathcal{S} = \mathsf{Unordered\text{-}trees}(\Sigma)$ (the other cases of trees have been covered by Proposition 3.2). Let $\mathcal{T}_1$ be the class of all ordered $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$-trees where $\Sigma_{\text{int}} = \Sigma_{\text{leaf}} = \Sigma$; then $\mathcal{T}_1$ is representation-feasible for $(\Sigma_{\text{rank}}, \rho)$ where $\Sigma_{\text{rank}} = \emptyset$ and ρ is the constant function of value 2. There is now a natural representation map $\mathsf{Str}_1 : \mathcal{T}_1 \to \mathcal{S}$ such that Str_1 "forgets" the ordering among the children of any node of its input tree. Using an MSO composition lemma for unordered trees (see Appendix A), we can see that Str_1 is an elementary MSO-good representation map, whereby using Theorem 4.2, we are done. Consider $\mathcal{S} = \mathsf{Nested\text{-}words}(\Sigma)$. For $\Sigma_{\text{leaf}} = \Sigma \cup (\Sigma \times \Sigma)$ and $\Sigma_{\text{int}} = (\Sigma \times \Sigma) \cup \{\circ\}$, if $\mathcal{T}_2$ is the class of all ordered $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$-trees whose leaf labels belong to Σ_{leaf} and internal node labels belong to Σ_{int}, then $\mathcal{T}_2$ is representation-feasible for $(\Sigma_{\text{rank}}, \rho)$ where again $\Sigma_{\text{rank}} = \emptyset$ and ρ is the constant function 2. There is then a natural map $\mathsf{Str}_2 : \mathcal{T}_2 \to \mathcal{S}$ as described for the example above. By an MSO composition lemma for nested words (see Appendix A), it follows that Str_2 is an elementary MSO-good representation map. We are then done by Theorem 4.2 again. The non-elementariness of witness functions is due to Theorem 4.2 and the non-elementary dependence on m, of the index of the $\equiv_{m,\mathcal{L}}$ relation over words [10]. ◀

B. n-partite cographs. The class of n-partite cographs, introduced in [12], can be defined up to isomorphism as the range of the map Str described as follows. Let $\Sigma_{\text{leaf}} = [n] = \{1, \ldots, n\}$ and $\Sigma_{\text{int}} = \{f \mid f : [n] \times [n] \to \{0, 1\}\}$. Let $\mathcal{T}$ be the class of all ordered $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$-trees whose leaf labels belong to Σ_{leaf} and internal node labels belong to Σ_{int}. Then $\mathcal{T}$ is representation-feasible for $(\Sigma_{\text{rank}}, \rho)$ where $\Sigma_{\text{rank}} = \emptyset$ and $\rho : \Sigma_{\text{int}} \to \mathbb{N}_+$ is the constant function 2. Consider $\mathsf{Str} : \mathcal{T} \to \mathsf{Graphs}$ defined as follows, where Graphs is the class of all undirected graphs: For $\mathsf{t} = (O, \lambda) \in \mathcal{T}$ where $O = ((A, \leq), \lesssim)$ is an ordered unlabeled tree and λ is the labeling function, $\mathsf{Str}(\mathsf{t}) = G = (V, E)$ is such that (i) V is exactly the set of leaf nodes of t (ii) for $a, b \in V$, if $c = a \wedge b$ is the greatest common ancestor (under $\leq$) of a and b in t, then $\{a, b\} \in E$ iff $\lambda(c)(\lambda(a), \lambda(b)) = 1$. We now have the following result. Below, a Σ-*labeled* n-partite cograph is a pair (G, ν) where G is an n-partite cograph and $\nu : V \to \Sigma$ is a labeling function. Also, "hereditary" means "closed under substructures".

▶ **Theorem 5.2.** *Given $n \in \mathbb{N}$ and a finite alphabet Σ, let $\mathsf{Labeled\text{-}n\text{-}partite\text{-}cographs}(\Sigma)$ be the class of all Σ-labeled n-partite cographs. Let $\mathcal{S}$ be any hereditary subclass of this class. Then $\mathcal{L}\text{-}\mathsf{EBSP}(\mathcal{S})$ holds with a computable witness function. Whereby, each of the graph classes below satisfies $\mathcal{L}\text{-}\mathsf{EBSP}(\cdot)$ with a computable witness function. Further, the classes with bounded parameters as mentioned below have elementary functions witnessing $\mathcal{L}\text{-}\mathsf{EBSP}(\cdot)$.*

1. *Any hereditary class of n-partite cographs, for each $n \in \mathbb{N}$.*
2. *Any hereditary class of graphs of bounded shrub-depth.*

3. *Any hereditary class of graphs of bounded $\mathcal{SC}$-depth.*
4. *Any hereditary class of graphs of bounded tree-depth.*
5. *Any hereditary class of cographs.*

Proof Idea. We first show the result for $\mathcal{S} = \mathsf{Labeled\text{-}n\text{-}partite\text{-}cographs}(\Sigma)$. The result for the various specific classes mentioned in the statement follows from the fact that $\mathcal{L}$-$\mathsf{EBSP}(\cdot)$ is closed under hereditary subclasses, and that all of the specific classes are hereditary subclasses of n-partite cographs [12]. Let $\Sigma_{\mathrm{leaf}} = [n] \times \Sigma$ and $\Sigma_{\mathrm{int}} = \{f \mid f : [n] \times [n] \to \{0, 1\}\}$. Then consider the class $\mathcal{T}$ of ordered $(\Sigma_{\mathrm{int}} \cup \Sigma_{\mathrm{leaf}})$-trees and the representation map $\mathsf{Str} : \mathcal{T} \to \mathsf{Labeled\text{-}n\text{-}partite\text{-}cographs}(\Sigma)$ exactly of the respective kinds described above for n-partite cographs. By an $\mathcal{L}$-composition lemma for labeled n-partite cographs (see Appendix A), we see that Str is elementary and $\mathcal{L}$-good, whereby we are done by Theorem 4.2. That the graph classes with the bounded parameters above have elementary witness functions follows again from Theorem 4.2 and elementary dependence on m, of the index of the $\equiv_{m,\mathcal{L}}$ relation over these classes (the latter follows from Theorem 3.2 of [11]). ◀

C. Classes generated using translation schemes. We look operations on classes of structures, that are "implementable" using quantifier-free translation schemes [23]. Given a vocabulary τ, let $\tau_{\text{disj-un},n}$ be the vocabulary obtained by expanding τ with n fresh unary predicates $P_1, \ldots, P_n$. Given τ-structures $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$ (assumed disjoint w.l.o.g.), the *n-disjoint sum of* $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$, denoted $\bigoplus_{i=1}^{i=n} \mathfrak{A}_i$, is the $\tau_{\text{disj-un},n}$-structure obtained upto isomorphism, by expanding the disjoint union $\bigsqcup_{i=1}^{i=n} \mathfrak{A}_i$ with $P_1, \ldots, P_n$ interpreted respectively as the universe of $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$. Let $\mathcal{S}_1, \ldots, \mathcal{S}_n$ be given classes of structures. A quantifier-free $(t, \tau_{\text{disj-un},n}, \tau, \mathrm{FO})$-translation scheme Ξ gives rise to an n-ary operation $\mathsf{O} : \mathcal{S}_1 \times \cdots \times \mathcal{S}_n \to \{\Xi(\bigoplus_{i=1}^{i=n} \mathfrak{A}_i) \mid \mathfrak{A}_i \in \mathcal{S}_i, 1 \le i \le n\}$ defined as $\mathsf{O}_1(\mathfrak{A}_1, \ldots, \mathfrak{A}_n) = \Xi(\bigoplus_{i=1}^{i=n} \mathfrak{A}_i)$. In this case, we say that O is *implementable using* Ξ. We say an operation is *quantifier-free*, if it is of the kind O just described. For a quantifier-free operation O, define the *dimension* of O to be the minimum of the dimensions of the quantifier-free translation schemes that implement O. We say O is "sum-like" if its dimension is one, else we say O is "product-like". We say O is *monotone* if any input of O is embeddable in the output of O. We say O *obeys $\mathcal{L}$-composition* if for each m, whenever an input of O is replaced with an $\mathcal{L}[m]$-equivalent input, the output of O is replaced with an $\mathcal{L}[m]$-equivalent output. The well-studied unary graph operations like complementation, transpose, and the line-graph operation, and binary operations like disjoint union and join are all sum-like, monotone and obey $\mathcal{L}$-composition. Likewise, the well-studied Cartesian, tensor, lexicographic, and strong products are all product-like, monotone and obey $\mathcal{L}$-composition. The central result of this section is as stated below. A proof sketch for this result in presented in Appendix B.

▶ **Theorem 5.3.** *Let $\mathcal{S}_1, \ldots, \mathcal{S}_n, \mathcal{S}$ be classes of structures and let $\mathsf{O} : \mathcal{S}_1 \times \cdots \times \mathcal{S}_n \to \mathcal{S}$ be a surjective n-ary quantifier-free operation. Then the following are true:*

1. *For each of the following cases, if $\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S}_i)$ holds (with computable/elementary witness functions) for each $i \in \{1, \ldots, n\}$, then $\mathcal{L}$-$\mathsf{EBSP}(\mathcal{S})$ holds as well (with computable/elementary witness functions):* (i) O *is sum-like* (ii) O *is product-like and $\mathcal{L} = FO$.*
2. *Suppose $\mathcal{S}_i$ admits an effective $\mathcal{L}$-good tree representation for each $i \in \{1, \ldots, n\}$, and* O *is monotone and obeys $\mathcal{L}$-composition. Then there exists an effective $\mathcal{L}$-good tree representation* $\mathsf{Str} : \mathcal{T} \to \mathcal{Z}$ *for the class $\mathcal{Z} = \mathcal{S} \cup \bigcup_{i=1}^{i=n} \mathcal{S}_i$.*
3. *Let* Str *be as given by the previous point. Then there is a linear time f.p.t. algorithm for* $\mathsf{MC}(\mathcal{L}, \mathcal{S})$ *that decides, for every $\mathcal{L}$ sentence φ (the parameter), if a given structure $\mathfrak{A}$ in $\mathcal{S}$ satisfies φ, provided that $\mathfrak{A}$ is given in the form of a tree representation of it under* Str.

▶ Discussion. Theorems 5.1, 5.2, 5.3 and 4.2 jointly show that the various posets and graph classes described in this section admit linear time f.p.t. algorithms for $\mathsf{MC}(\mathcal{L}, \cdot)$, provided an $\mathcal{L}$-good tree representation of the input structure is given. For structures coming from the classes of words, the various kinds of trees, nested words and cographs, we can indeed even construct, in linear time, the Hasse diagram of an $\mathcal{L}$-good tree representation for the structure, given its standard presentation (this is easy to see for the first three kinds of classes; for the case of cographs, see [18]). It turns out that the techniques used in our f.p.t. algorithms can be easily adapted to work even when the input structures are presented using the aforementioned diagrams. This then enables getting (unconditional) linear time f.p.t. algorithms for $\mathsf{MC}(\mathcal{L}, \cdot)$ for the cases of words, trees, nested words and cographs, thereby matching known f.p.t. results for $\mathsf{MC}(\mathcal{L}, \cdot)$ concerning these classes [10, 2, 19]. Going further, to the best of our knowledge, the f.p.t. results for n-partite cographs and those for classes generated using trees of quantifier-free operations, that are entailed by Theorems 5.2, 5.3 and 4.2, are new. Our proofs can then be seen as giving a different and unified technique to show existing f.p.t. results, in addition to giving new results. We mention however that if the dependence on the parameter in our f.p.t. algorithms is also considered, then our results (which give only computable parameter dependence) are weaker than those in [11] which show that for classes of bounded tree-depth/$\mathcal{SC}$-depth/shrub-depth, there are linear time f.p.t. algorithms for $\mathcal{L}$ model checking, that have elementary parameter dependence.

6 Logical fractals

We define a strengthening of $\mathcal{L}$-EBSP that asserts "logical self-similarity" at "all scales" for a suitable notion of scale. Towards the formal definition, call a function $f : \mathbb{N}_+ \to \mathbb{N}_+$ a *scale function* if it is strictly increasing. The i^{th} *scale*, denoted $\langle i \rangle_f$, is defined as the interval $[f(i-1)+1, f(i)] = \{j \mid f(i-1)+1 \leq j \leq f(i)\}$ for $i > 1$, and $[1, f(1)] = \{j \mid 1 \leq j \leq f(1)\}$.

▶ **Definition 6.1** (Logical fractal). Given a class $\mathcal{S}$ of structures, we say $\mathcal{S}$ is an $\mathcal{L}$-*fractal*, if there exists a function $\theta_{(\mathcal{S},\mathcal{L})} : \mathbb{N}_+^2 \to \mathbb{N}_+$ such that (i) $\theta_{(\mathcal{S},\mathcal{L})}(m)$ is a scale function for all $m \in \mathbb{N}$, and (ii) for each structure $\mathfrak{A}$ of $\mathcal{S}$ and each $m \in \mathbb{N}$, if f is the function $\theta_{(\mathcal{S},\mathcal{L})}(m)$ and $|\mathfrak{A}| \in \langle i \rangle_f$ for some $i \in \mathbb{N}$, then for all $j < i$, there exists a substructure $\mathfrak{B}$ of $\mathfrak{A}$ in $\mathcal{S}$, such that $|\mathfrak{B}| \in \langle j \rangle_f$ and $\mathfrak{B} \equiv_{m,\mathcal{L}} \mathfrak{A}$. We say $\theta_{(\mathcal{S},\mathcal{L})}$ is a *witness* to the $\mathcal{L}$-fractal property of $\mathcal{S}$.

Call a representation map $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$ as $\mathcal{L}$-*great* if (i) it is $\mathcal{L}$-good, and (ii) there is a strictly increasing function $\beta : \mathbb{N} \to \mathbb{N}$ such that for every $\mathsf{t}, \mathsf{s} \in \mathcal{T}$, if $|(|\mathsf{t}| - |\mathsf{s}|)| \leq n$, then $|(|\mathsf{Str}(\mathsf{t})| - |\mathsf{Str}(\mathsf{s})|)| \leq \beta(n)$. In such a case, we say $\mathcal{S}$ *admits* an $\mathcal{L}$-great tree representation. We now have the following result. We present a proof sketch in Appendix C.

▶ **Proposition 6.2.** *If $\mathcal{S}$ admits an $\mathcal{L}$-great tree representation, then $\mathcal{S}$ is an $\mathcal{L}$-fractal.*

One can easily verify that for all the examples of posets and graphs considered in Section 5, their $\mathcal{L}$-good representation maps as described in this section are indeed $\mathcal{L}$-great too, whereby all these classes are logical fractals. Further, the logical fractal property is preserved under all examples of operations on structures seen in Section 5; see Appendix C for a proof sketch.

7 Conclusion

We presented a natural finitary analogue of the well-studied downward Löwenheim-Skolem property from classical model theory, denoted $\mathcal{L}$-EBSP, and showed that this property is enjoyed by various classes of interest in computer science, whereby all these classes can be seen to admit a natural finitary version of the downward Löwenheim-Skolem theorem. The aforesaid classes admit tree representations for their structures, using which we obtain linear time f.p.t. algorithms for FO and MSO model checking for these classes (when the structures in the classes are presented using their tree representations). Finally, the aforesaid classes possess a fractal like property, one based on logic. These observations open up several interesting and challenging directions for future work, some of which we mention below.

1. Under what conditions on a class of structures is the index of the $\equiv_{m,\mathcal{L}}$ relation over the class an elementary function of m? Investigating this question for classes admitting elementary $\mathcal{L}$-good tree representations might yield insights for linear time f.p.t. algorithms for $\mathcal{L}$ model checking over these classes, that have elementary parameter dependence.
2. We have adopted the composition method for proving f.p.t. results for the classes we have considered. However, automata based methods for f.p.t. results have also been well-studied in the literature [6, 14, 11]. Since all our classes have representations using trees, a natural question is whether we can give tree automata based proofs for them, and more generally for Theorem 4.2. Given the existence of pumping lemmas for trees, the aforementioned proofs would also, intuitively speaking, lend themselves to investigating the *upward* Löwenheim-Skolem property for all of our classes.
3. The requirement of a small and logically similar *substructure* in the $\mathcal{L}$-EBSP definition, causes some well-studied classes of structures to not satisfy $\mathcal{L}$-EBSP. For instance, the class of all graphs of tree-width $\leq k$ does not satisfy $\mathcal{L}$-EBSP for any k. This motivates asking whether these classes satisfy variants of $\mathcal{L}$-EBSP in which "substructure" is replaced with other natural relations, and if they do, then whether the witness functions are computable. As a step in this direction, we show that the variant of $\mathcal{L}$-EBSP in which "substructure" is replaced with "minor", is satisfied by the class of all graphs, and every minor-hereditary subclass of it. Any witness function however, turns out to be non-recursive in general.
4. All our examples of $\mathcal{L}$-EBSP classes (resp. logical fractals) are defined using structural conditions. This leads us to asking the converse, and hence the following: Is there a structural characterization of $\mathcal{L}$-EBSP (resp. of logical fractals)? We believe that an answer to this question, even under reasonable assumptions, would yield new classes that are well-behaved from both the logical and the algorithmic perspectives.

Acknowledgements. I express my deepest gratitude to Bharat Adsul for various insightful discussions and critical feedback that have helped in preparing this paper. I also thank the anonymous referees of the paper, and of an earlier version of it, for their valuable comments.

References

1 Natasha Alechina and Yuri Gurevich. Syntax vs. semantics on finite structures. In *Structures in Logic and Computer Science. A Selection of Essays in Honor of A. Ehrenfeucht*, pages 14–33. Springer-Verlag, 1997.
2 Rajeev Alur and Parthasarathy Madhusudan. Adding nesting structure to words. *J. ACM*, 56(3), 2009.
3 Albert Atserias, Anuj Dawar, and Martin Grohe. Preservation under extensions on well-behaved finite structures. *SIAM J. Comput.*, 38(4):1364–1381, 2008. doi:10.1137/060658709.
4 Michael Barnsley. *Fractals Everywhere.* Academic Press Professional, Inc., 1988.
5 Hubert Comon, Max Dauchet, Remi Gilleron, Christof Löding, Florent Jacquemard, Denis Lugiez, Sophie Tison, and Marc Tommasi. Tree automata techniques and applications, 2007. release October 12, 2007. URL: http://www.grappa.univ-lille3.fr/tata.

6 Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.

7 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000.

8 Michael Elberfeld, Martin Grohe, and Till Tantau. Where first-order and monadic second-order logic coincide. In *LICS 2012, Croatia, June 25-28, 2012*, pages 265–274, 2012.

9 Solomon Feferman and Robert Vaught. The first order properties of products of algebraic systems. *Fundamenta Mathematicae*, 47(1):57–103, 1959.

10 Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Logic*, 130(1-3):3–31, 2004.

11 Jakub Gajarsky and Petr Hlinený. Kernelizing MSO properties of trees of fixed height, and some consequences. *Log. Meth. Comp. Sci.*, 11(19):1–26, 2015.

12 Robert Ganian, Petr Hlinený, Jaroslav Nešetřil, Jan Obdrzálek, Patrice Ossona de Mendez, and Reshma Ramadurai. When trees grow low: Shrubs and fast MSO1. In *MFCS 2012, Bratislava, Slovakia, August 27-31, 2012*, pages 419–430, 2012.

13 Martin Grohe. Some remarks on finite Löwenheim-Skolem theorems. *Math. Log. Q.*, 42:569–571, 1996.

14 Martin Grohe. Logic, graphs, and algorithms. *Logic and automata*, 2:357–422, 2008.

15 Martin Grohe and Stephan Kreutzer. Methods for algorithmic meta theorems. *Model Theoretic Methods in Finite Combinatorics*, 558:181–206, 2011.

16 Yuri Gurevich. Toward logic tailored for computational complexity. In Michael M. Richter et al., editor, *Computation and Proof Theory: Proceedings of the Logic Colloquium held in Aachen, July 18-23, 1983, Part II*, pages 175–216. Springer-Verlag, 1984.

17 Frederik Harwath, Lucas Heimberg, and Nicole Schweikardt. Preservation and decomposition theorems for bounded degree structures. *Log. Meth. Comp. Sci.*, 11(4), 2015.

18 Beverly Jamison and Stephan Olariu. Recognizing P_4-sparse graphs in linear time. *SIAM Journal on Computing*, 21(2):381–406, 1992.

19 Beverly Jamison and Stephan Olariu. Linear time optimization algorithms for P_4-sparse graphs. *Discrete Applied Mathematics*, 61(2):155–175, 1995.

20 Leonid Libkin. *Elements of Finite Model Theory.* Springer-Verlag, 2004.

21 Per Lindström. A characterization of elementary logic. In Sören Halldén, editor, *Modality, Morality and Other Problems of Sense and Nonsense*, pages 189–191. Lund,Gleerup, 1973.

22 Leopold Löwenheim. Über möglichkeiten im relativkalkül. *Mathematische Annalen*, 76(4):447–470, 1915.

23 Johann A. Makowsky. Algorithmic uses of the Feferman-Vaught theorem. *Ann. Pure Appl. Logic*, 126(1-3):159–213, 2004.

24 Anatoly I. Maltsev. Untersuchungen aus dem Gebiete der mathematischen Logik. *Matematicheskii Sbornik*, n.s.(1):323–336, 1936.

25 Jaroslav Nešetřil and Patrice Ossona de Mendez. Tree-depth, subgraph coloring and homomorphism bounds. *Eur. J. Comb.*, 27(6):1022–1041, 2006.

26 Eric Rosen. Some aspects of model theory and finite structures. *Bull. Symbolic Logic*, 8(3):380–403, 2002.

27 Benjamin Rossman. Homomorphism preservation theorems. *J. ACM*, 55(3):15:1–15:53, 2008.

28 Abhisekh Sankaran. A generalization of the Łoś-Tarski preservation theorem. *CoRR*, abs/1609.06297, 2016. URL: `http://arxiv.org/abs/1609.06297`.

29 Abhisekh Sankaran. A finitary analogue of the downward Löwenheim-Skolem property. *CoRR*, abs/1705.04493, 2017. URL: `http://arxiv.org/abs/1705.04493`.

30 Abhisekh Sankaran, Bharat Adsul, and Supratik Chakraborty. A generalization of the Łoś-Tarski preservation theorem over classes of finite structures. In *MFCS 2014, Budapest, Hungary, August 25-29, 2014, Part I*, pages 474–485, 2014.

31 Abhisekh Sankaran, Bharat Adsul, and Supratik Chakraborty. A generalization of the Łoś-Tarski preservation theorem. *Ann. Pure Appl. Logic*, 167(3):189–210, 2016.

32 J. W. Thatcher and J. B. Wright. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical systems theory*, 2(1):57–81, 1968.

33 Wolfgang Thomas. *Ehrenfeucht games, the composition method, and the monadic theory of ordinal words*, pages 118–143. Springer Berlin Heidelberg, 1997.

34 Jouko Väänänen. Pseudo-finite model theory. *Mat. Contemp*, 24(8th):169–183, 2003.

A Feferman-Vaught style composition lemmas

We present Feferman-Vaught style composition lemmas for various classes of structures. While these lemmas for ordered and unordered trees, as presented here, is possibly known, to the best of our knowledge these lemmas for nested words and n-partite cographs (again as presented here) are new. The proofs of all these lemmas can be found in [29].

A. Ordered trees. To state the composition lemma, we first introduce some terminology. For a finite alphabet Ω, given ordered Ω-trees t, s having disjoint sets of nodes (w.l.o.g.) and a non-root node a of t, the *join of* s *to* t *to the right of* a, denoted $\mathsf{t} \cdot_a^{\rightarrow} \mathsf{s}$, is defined as the tree obtained by making s as a new child subtree of the parent of a in t, at the successor position of the position of a among the children of the parent of a in t. We can similarly define the *join of* s *to* t *to the left of* a, denoted $\mathsf{t} \cdot_a^{\leftarrow} \mathsf{s}$. Likewise, for t and s as above, if a is a leaf node of t, we can define the *join of* s *to* t *below* a, denoted $\mathsf{t} \cdot_a^{\uparrow} \mathsf{s}$, as the tree obtained up to isomorphism by making the root of s as a child of a.

▶ **Lemma A.1** (Composition lemma for ordered trees). *For a finite alphabet Ω, let $\mathsf{t}_i, \mathsf{s}_i$ be non-empty ordered Ω-trees, and let a_i be a non-root node of t_i, for $i \in \{1, 2\}$. Let $m \geq 2$ and suppose that $(\mathsf{t}_1, a_1) \equiv_{m,\mathcal{L}} (\mathsf{t}_2, a_2)$ and $\mathsf{s}_1 \equiv_{m,\mathcal{L}} \mathsf{s}_2$. Then each of the following hold.*
1. $((\mathsf{t}_1 \cdot_{a_1}^{\rightarrow} \mathsf{s}_1), a_1) \equiv_{m,\mathcal{L}} ((\mathsf{t}_2 \cdot_{a_2}^{\rightarrow} \mathsf{s}_2), a_2)$
2. $((\mathsf{t}_1 \cdot_{a_1}^{\leftarrow} \mathsf{s}_1), a_1) \equiv_{m,\mathcal{L}} ((\mathsf{t}_2 \cdot_{a_2}^{\leftarrow} \mathsf{s}_2), a_2)$
3. $((\mathsf{t}_1 \cdot_{a_1}^{\uparrow} \mathsf{s}_1), a_1) \equiv_{m,\mathcal{L}} ((\mathsf{t}_2 \cdot_{a_2}^{\uparrow} \mathsf{s}_2), a_2)$ *if a_1, a_2 are leaf nodes of $\mathsf{t}_1, \mathsf{t}_2$ resp.*

B. Unordered trees. We introduce terminology akin to that introduced for ordered trees above. Given unordered trees t and s, and a node a of t, define the *join of* s *to* t *at* a, denoted $\mathsf{t} \cdot_a \mathsf{s}$, as follows: Let s' be an isomorphic copy of s whose set of nodes is disjoint with the set of nodes of t. Then $\mathsf{t} \cdot_a \mathsf{s}$ is defined up to isomorphism as the tree obtained by making s' as a new child subtree of a in t.

▶ **Lemma A.2** (Composition lemma for unordered trees). *For a finite alphabet Ω, let $\mathsf{t}_i, \mathsf{s}_i$ be non-empty unordered Ω-trees, and let a_i be a node of t_i, for $i \in \{1, 2\}$. For $m \in \mathbb{N}$, suppose that $(\mathsf{t}_1, a_1) \equiv_{m,\mathcal{L}} (\mathsf{t}_2, a_2)$ and $\mathsf{s}_1 \equiv_{m,\mathcal{L}} \mathsf{s}_2$. Then $((\mathsf{t}_1 \cdot_{a_1} \mathsf{s}_1), a_1) \equiv_{m,\mathcal{L}} ((\mathsf{t}_2 \cdot_{a_2} \mathsf{s}_2), a_2)$.*

C. Nested words. We first define the notion of *insert of a nested word* v *in a nested word* u *at a given position* e *of* u.

▶ **Definition A.3** (Insert). Let $\mathsf{u} = (A_{\mathsf{u}}, \leq_{\mathsf{u}}, \lambda_{\mathsf{u}}, \rightsquigarrow_{\mathsf{u}})$ and $\mathsf{v} = (A_{\mathsf{v}}, \leq_{\mathsf{v}}, \lambda_{\mathsf{v}}, \rightsquigarrow_{\mathsf{v}})$ be given nested Σ-words, and let e be a position in u. The *insert of* v *in* u *at* e, denoted $\mathsf{u} \uparrow_e \mathsf{v}$, is a nested Σ-word defined as below.

1. If u and v have disjoint sets of positions, then $\mathsf{u} \uparrow_e \mathsf{v} = (A, \leq, \lambda, \rightsquigarrow)$ where
   - $A = A_{\mathsf{u}} \sqcup A_{\mathsf{v}}$
   - $\leq = \leq_{\mathsf{u}} \cup \leq_{\mathsf{v}} \cup \{(i,j) \mid i \in A_{\mathsf{u}}, j \in A_{\mathsf{v}}, i \leq_{\mathsf{u}} e\} \cup \{(j,i) \mid i \in A_{\mathsf{u}}, j \in A_{\mathsf{v}}, e \leq_{\mathsf{u}} i, e \neq i\}$
   - $\lambda(a) = \lambda_{\mathsf{u}}(a)$ if $a \in A_{\mathsf{u}}$, else $\lambda(a) = \lambda_{\mathsf{v}}(a)$
   - $\rightsquigarrow = \rightsquigarrow_{\mathsf{u}} \cup \rightsquigarrow_{\mathsf{v}}$
2. If u and v have overlapping sets of positions, then let v_1 be an isomorphic copy of v whose set of positions is disjoint with that of u. Then $\mathsf{u} \uparrow_e \mathsf{v}$ is defined up to isomorphism as $\mathsf{u} \uparrow_e \mathsf{v}_1$.

In the special case that e is the last (under $\leq_{\mathsf{u}}$) position of u, we denote $\mathsf{u} \uparrow_e \mathsf{v}$ as $\mathsf{u} \cdot \mathsf{v}$, and call the latter as the *concatenation of* v *with* u.

▶ **Lemma A.4** (Composition lemma for nested words). *For a finite alphabet Σ, let $\mathsf{u}_i, \mathsf{v}_i \in$ Nested-words(Σ), and let e_i be a position in u_i for $i \in \{1,2\}$. Then the following hold for each $m \in \mathbb{N}$.*
1. *If $(\mathsf{u}_1, e_1) \equiv_{m,\mathcal{L}} (\mathsf{u}_2, e_2)$ and $\mathsf{v}_1 \equiv_{m,\mathcal{L}} \mathsf{v}_2$, then $(\mathsf{u}_1 \uparrow_{e_1} \mathsf{v}_1) \equiv_{m,\mathcal{L}} (\mathsf{u}_2 \uparrow_{e_2} \mathsf{v}_2)$.*
2. *$\mathsf{u}_1 \equiv_{m,\mathcal{L}} \mathsf{u}_2$ and $\mathsf{v}_1 \equiv_{m,\mathcal{L}} \mathsf{v}_2$, then $\mathsf{u}_1 \cdot \mathsf{v}_1 \equiv_{m,\mathcal{L}} \mathsf{u}_2 \cdot \mathsf{v}_2$.*

D. n-partite cographs. Let $\mathcal{T}$ and Str be as described in the proof idea of Theorem 5.2. Specifically, $\mathcal{T}$ is the class of all $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$-trees whose leaf labels belong to Σ_{leaf} and internal node labels belong to Σ_{int}, where $\Sigma_{\text{leaf}} = [n] \times \Sigma$ and $\Sigma_{\text{int}} = \{f \mid f : [n] \times [n] \to \{0,1\}\}$. The representation map Str : $\mathcal{T} \to$ Labeled-n-partite-cographs(Σ) is exactly of the kind described for n-partite cographs, that maps a tree in $\mathcal{T}$ to the labeled n-partite graph that it represents.

▶ **Lemma A.5** (Composition lemma for n-partite cographs). *For $i \in \{1,2\}$, let $(G_i, \nu_{i,1})$ and $(H_i, \nu_{i,2})$ be graphs in Labeled-n-partite-cographs(Σ). Suppose t_i and s_i are trees of $\mathcal{T}$ such that Str$(\mathsf{t}_i) = (G_i, \nu_{i,1})$, Str$(\mathsf{s}_i) = (H_i, \nu_{i,2})$, and the labels of root$(\mathsf{t}_i)$ and root(s_i) are the same. Let $\mathsf{z}_i = \mathsf{t}_i \odot \mathsf{s}_i$ and Str$(\mathsf{z}_i) = (Z_i, \nu_i)$ for $i \in \{1,2\}$. For each $m \in \mathbb{N}$, if $(G_1, \nu_{1,1}) \equiv_{m,\mathcal{L}} (G_2, \nu_{2,1})$ and $(H_1, \nu_{1,2}) \equiv_{m,\mathcal{L}} (H_2, \nu_{2,2})$, then $(Z_1, \nu_1) \equiv_{m,\mathcal{L}} (Z_2, \nu_2)$.*

B Proof sketch for Theorem 5.3

Recall from Section 2 that a $(t, \tau, \nu, \mathcal{L})$-translation scheme Ξ defines a map from τ-structures to ν-structures, that we denote by Ξ again. For a class $\mathcal{S}$ of τ-structures, we let $\Xi(\mathcal{S})$ denote the class $\{\Xi(\mathfrak{A}) \mid \mathfrak{A} \in \mathcal{S}\}$. Let n-disjoint-sum$(\mathcal{S}_1, \ldots, \mathcal{S}_n) = \{\bigoplus_{i=1}^{i=n} \mathfrak{A}_i \mid \mathfrak{A}_i \in \mathcal{S}_i, 1 \leq i \leq n\}$, where $\bigoplus_{i=1}^{i=n} \mathfrak{A}_i$ denotes the n-disjoint sum of $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$. Call a quantifier-free translation scheme *scalar* if its dimension is one. We now have the following three results that enable us to prove Theorem 5.3. The proofs of these results can be found in [29].

▶ **Lemma B.1.** *Let $\mathcal{S}$ be a class of τ-structures and Ξ be a quantifier-free (t, τ, ν, FO)-translation scheme. Given structures $\mathfrak{A}$ and $\mathfrak{B}$ from $\mathcal{S}$, if $\mathfrak{B} \subseteq \mathfrak{A}$, then $\Xi(\mathfrak{B}) \subseteq \Xi(\mathfrak{A})$.*

▶ **Lemma B.2.** *Let $\mathcal{S}, \mathcal{S}_1, \ldots, \mathcal{S}_n$ be classes of structures for $n \geq 1$. If $\mathcal{L}$-EBSP$(\mathcal{S}_i)$ is true for each $i \in \{1, \ldots, n\}$, then so is $\mathcal{L}$-EBSP$(n$-disjoint-sum$(\mathcal{S}_1, \ldots, \mathcal{S}_n))$. Further, if there is a computable/elementary witness function for $\mathcal{L}$-EBSP$(\mathcal{S}_i)$ for each $i \in \{1, \ldots, n\}$, then there is a computable/elementary witness function for $\mathcal{L}$-EBSP$(n$-disjoint-sum$(\mathcal{S}_1, \ldots, \mathcal{S}_n))$ as well.*

▶ **Proposition B.3.** *Let $\mathcal{S}$ be class of τ-structures and Ξ be a quantifier-free (t, τ, ν, FO)-translation scheme. Then the following are true:*
1. *If FO-EBSP$(\mathcal{S})$ is true, then so is FO-EBSP$(\Xi(\mathcal{S}))$.*
2. *If Ξ is scalar and MSO-EBSP$(\mathcal{S})$ is true, then so is MSO-EBSP$(\Xi(\mathcal{S}))$.*

In each of the implications above, a computable/elementary witness function for the antecedent implies a computable/elementary witness function for the consequent.

Proof of Theorem 5.3.

(1) Follows easily from Lemma B.2 and Proposition B.3.

(2) Let $\mathsf{Str}_i : \mathcal{T}_i \to \mathcal{S}_i$ be an effective $\mathcal{L}$-good representation map for $1 \leq i \leq n$, where $\mathcal{T}_i$ is a class of trees over $(\Sigma^i_{\text{int}} \cup \Sigma^i_{\text{leaf}})$ that is representation feasible for $(\Sigma^i_{\text{rank}}, \rho_i)$.
Let O be a new label that is not in $(\Sigma^i_{\text{int}} \cup \Sigma^i_{\text{leaf}})$ for any $i \in \{1, \ldots, n\}$. Define $\Sigma_{\text{int}}, \Sigma_{\text{leaf}}, \Sigma_{\text{rank}}$ and $\rho : \Sigma_{\text{int}} \to \mathbb{N}_+$ as follows:
- $\Sigma_{\text{int}} = \{O\} \cup \bigcup_{i=1}^{i=n} \Sigma^i_{\text{int}}$
- $\Sigma_{\text{leaf}} = \bigcup_{i=1}^{i=n} \Sigma^i_{\text{leaf}}$
- $\Sigma_{\text{rank}} = \{O\} \cup \bigcup_{i=1}^{i=n} \Sigma^i_{\text{rank}}$
- $\rho = \{(O, n)\} \cup \bigcup_{i=1}^{i=n} \rho_i$.

Let $\widehat{\mathcal{T}}$ be the class of all trees over $(\Sigma_{\text{int}} \cup \Sigma_{\text{leaf}})$ obtained by taking $\mathsf{t}_i \in \mathcal{T}_i$ for $1 \leq i \leq n$, and making $\mathsf{t}_1, \ldots, \mathsf{t}_n$ as child subtrees (and in that order) of a new root node whose label is O. Let $\mathcal{T} = \widehat{\mathcal{T}} \cup \bigcup_{i=1}^{i=n} \mathcal{T}_i$. Verify that $\mathcal{T}$ is indeed representation feasible for $(\Sigma_{\text{rank}}, \rho)$. Let $\mathsf{Str} : \mathcal{T} \to \mathcal{Z}$ be such that for $\mathsf{t} \in \mathcal{T}$, if $\mathsf{t} \in \mathcal{T}_i$, then $\mathsf{Str}(\mathsf{t}) = \mathsf{Str}_i(\mathsf{t})$. Else, let $a_1, \ldots, a_n$ be the children of the root of t. Then by the construction of $\mathcal{T}$, we have $\mathsf{t}_{\geq a_i} \in \mathcal{T}_i$ and that the label of the root of t is O. Define $\mathsf{Str}(\mathsf{t}) = \mathsf{O}(\mathsf{Str}_1(\mathsf{t}_{\geq a_1}), \ldots, \mathsf{Str}_n(\mathsf{t}_{\geq a_n}))$. Using the fact that O is monotone and obeys $\mathcal{L}$-composition, and using Lemma B.1, it is easy to verify that Str is indeed an effective $\mathcal{L}$-good representation map.

(3) Since Str is effective and $\mathcal{L}$-good, by Theorem 4.2, there is a linear time f.p.t. algorithm for $\mathsf{MC}(\mathcal{L}, \mathcal{Z})$ that decides, for every $\mathcal{L}$ sentence φ, if a given structure $\mathfrak{A}$ in $\mathcal{Z}$ satisfies φ, provided that $\mathfrak{A}$ is given in the form of a tree representation of it under Str. Clearly the same algorithm is also f.p.t. for $\mathsf{MC}(\mathcal{L}, \mathcal{S})$. ◀

C Proof sketches for results concerning logical fractals

A. Proof sketch for Proposition 6.2. The following lemma is central to the proof of Proposition 6.2. The proof of the lemma uses similar ideas as used in proving Theorem 4.2; The details are presented in [29].

▶ **Lemma C.1.** *Let $\mathcal{S}$ be a class of structures that admits an $\mathcal{L}$-good tree representation $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$. Then there exists a strictly increasing computable function $\eta : \mathbb{N} \to \mathbb{N}$ such that for each $m \in \mathbb{N}$ and for each tree $\mathsf{t} \in \mathcal{T}$ of size $> \eta(m)$, there exists a proper subtree s of t in $\mathcal{T}$ such that* (i) $\mathsf{Str}(\mathsf{s}) \hookrightarrow \mathsf{Str}(\mathsf{t})$, (ii) $\mathsf{Str}(\mathsf{s}) \equiv_{m,\mathcal{L}} \mathsf{Str}(\mathsf{t})$, *and* (iii) $|\mathsf{t}| - |\mathsf{s}| \leq \eta(m)$.

Proof of Proposition 6.2. Let $\mathsf{Str} : \mathcal{T} \to \mathcal{S}$ be an $\mathcal{L}$-great representation. Then Str is $\mathcal{L}$-good, whereby by Lemma C.1, there exists a function η satisfying the properties mentioned in the lemma. For $m \in \mathbb{N}$, define $f(m) = \max\{|\mathsf{Str}(\mathsf{t})| \mid |\mathsf{t}| \leq \eta(m)\}$. Since Str is $\mathcal{L}$-great, there exists a function $\beta : \mathbb{N} \to \mathbb{N}$ satisfying the properties mentioned in the definition of $\mathcal{L}$-greatness. Then define the function $\theta_{(\mathcal{S},\mathcal{L})} : \mathbb{N}^2_+ \to \mathbb{N}_+$ as $\theta_{(\mathcal{S},\mathcal{L})}(m)(n) = f(m) + (n-1) \cdot \beta(\eta(m))$. It is easily seen that $\theta_{(\mathcal{S},\mathcal{L})}(m)$ is a scale function. Consider $\mathfrak{A} \in \mathcal{S}$ and $m \in \mathbb{N}$, and suppose that $|\mathfrak{A}| \in \langle i \rangle_g$ where g is the function $\theta_{(\mathcal{S},\mathcal{L})}(m)$ and $i > 1$. To show that for $j < i$, there exists a substructure $\mathfrak{B}$ of $\mathfrak{A}$ in $\mathcal{S}$ such that $|\mathfrak{B}| \in \langle j \rangle_g$ and $\mathfrak{B} \equiv_{m,\mathcal{L}} \mathfrak{A}$, we observe that it suffices to show the same simply for $j = i - 1$. Let $\mathsf{t} \in \mathcal{T}$ be such that $\mathsf{Str}(\mathsf{t}) = \mathfrak{A}$. By Lemma C.1, there exists a subtree s of t in $\mathcal{T}$ such that $\mathsf{Str}(\mathsf{s}) \hookrightarrow \mathsf{Str}(\mathsf{t})$, $\mathsf{Str}(\mathsf{s}) \equiv_{m,\mathcal{L}} \mathsf{Str}(\mathsf{t})$ and $|\mathsf{t}| - |\mathsf{s}| \leq \eta(m)$. Since Str is $\mathcal{L}$-great, it follows that $|\mathsf{Str}(\mathsf{t})| - |\mathsf{Str}(\mathsf{s})| \leq \beta(\eta(m))$. Whereby, either $|\mathsf{Str}(\mathsf{s})| \in \langle i-1 \rangle_g$ or $|\mathsf{Str}(\mathsf{s})| \in \langle i \rangle_g$. If the former holds, then taking $\mathfrak{B} = \mathsf{Str}(\mathsf{s})$, we are done. If the latter holds, then applying Lemma C.1 recursively to s, we eventually get a subtree x of t in $\mathcal{T}$ such that $\mathsf{Str}(\mathsf{x}) \hookrightarrow \mathsf{Str}(\mathsf{t})$, $\mathsf{Str}(\mathsf{x}) \equiv_{m,\mathcal{L}} \mathsf{Str}(\mathsf{t})$ and $|\mathsf{Str}(\mathsf{x})| \in \langle i-1 \rangle_g$. Whereby, taking $\mathcal{B} = \mathsf{Str}(\mathsf{x})$, we are done. ◀

B. Proof sketch for closure properties. We finally show that the logical fractal property remains closed under the examples of operations seen earlier: the unary operations of complementation, transpose and the line-graph operation, the binary sum-like operations of disjoint union and join, and the binary product-like operations of Cartesian, tensor, strong and lexicographic products. The proofs of each of these are along the same lines as the corresponding proofs that show the closure of $\mathcal{L}$-EBSP under these operations, as given by Theorem 5.3(1). We give below the witness functions for each of these operations.

For $i \in \{1,2\}$, let $\mathcal{S}_i$ be an $\mathcal{L}$-fractal and let $\theta_{(\mathcal{S}_i,\mathcal{L})}$ be a witness to the $\mathcal{L}$-fractal property of $\mathcal{S}_i$. Let O be one of the operations mentioned above, and let $\mathcal{S}$ be the class $\{O(\mathfrak{A}) \mid \mathfrak{A} \in \mathcal{S}_1\}$ if O is unary, and the class $\{O(\mathfrak{A},\mathfrak{B}) \mid \mathfrak{A} \in \mathcal{S}_1, \mathfrak{B} \in \mathcal{S}_2\}$ if O is binary. Then $\mathcal{S}$ is an $\mathcal{L}$-fractal with witness function $\theta_{(\mathcal{S},\mathcal{L})}$ given as follows:

1. O is unary:
   a. If O is complementation or transpose, then $\theta_{(\mathcal{S},\mathcal{L})} = \theta_{(\mathcal{S}_1,\mathcal{L})}$.
   b. If O is the line-graph operation and $\mathcal{L} = \text{FO}$, then $\theta_{(\mathcal{S},\mathcal{L})}(m)(n) = (\theta_{(\mathcal{S}_1,\mathcal{L})}(m)(n))^2$ for all $m,n \in \mathbb{N}_+$.
2. O is binary and sum-like: $\theta_{(\mathcal{S},\mathcal{L})}(m)(n) = \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(1) + \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(n)$ for all $m,n \in \mathbb{N}_+$.
3. O is binary and product-like, and $\mathcal{L} = \text{FO}$: We define $\theta_{(\mathcal{S},\mathcal{L})}(m)(n)$ inductively as follows for all $m,n \in \mathbb{N}_+$:
   - $\theta_{(\mathcal{S},\mathcal{L})}(m)(1) = \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(1) \cdot \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(1)$
   - Let $n > 1$. For $i \in \{1,2\}$, let $k_i = \min(\{j \mid \theta_{(\mathcal{S}_i,\mathcal{L})}(m)(j) \geq \theta_{(\mathcal{S},\mathcal{L})}(m)(n)\})$. Then $\theta_{(\mathcal{S},\mathcal{L})}(m)(n+1) = \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(k_1+1) \cdot \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(k_2+1)$.

For unary and sum-like binary operations, it is easy to verify that $\theta_{(\mathcal{S},\mathcal{L})}$ indeed witnesses the $\mathcal{L}$-fractal property of $\mathcal{S}$. For product-like binary operations, here is the explanation. For $m \in \mathbb{N}_+$, let f be the function $\theta_{(\mathcal{S},\mathcal{L})}(m)$. Consider $\mathfrak{C} \in \mathcal{S}$ such that $|\mathfrak{C}| \in \langle i \rangle_f$. We assume $i > 2$; the $i = 2$ case can be done similarly. Then $\mathfrak{C} = O(\mathfrak{A},\mathfrak{B})$ for $\mathfrak{A} \in \mathcal{S}_1$ and $\mathfrak{B} \in \mathcal{S}_2$. By construction of $\theta_{(\mathcal{S},\mathcal{L})}$, we have $|\mathfrak{C}| = |\mathfrak{A}| \cdot |\mathfrak{B}| \geq f(i-1) = \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(k_1+1) \cdot \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(k_2+1)$, where $k_i = \min(\{j \mid \theta_{(\mathcal{S}_i,\mathcal{L})}(m)(j) \geq f(i-2)\})$ for $i \in \{1,2\}$. We have two cases here:

1. $|\mathfrak{A}| \geq \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(k_1+1)$ and $|\mathfrak{B}| \geq \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(k_2+1)$: Then since $\mathcal{S}_i$ is an $\mathcal{L}$-fractal with witness $\theta_{(\mathcal{S}_i,\mathcal{L})}$ for $i \in \{1,2\}$, there exists a substructure $\mathfrak{A}'$ of $\mathfrak{A}$ in $\mathcal{S}_1$, resp. substructure $\mathfrak{B}'$ of $\mathfrak{B}$ in $\mathcal{S}_2$ such that $\mathfrak{A}' \equiv_{m,\mathcal{L}} \mathfrak{A}$ and $\theta_{(\mathcal{S}_1,\mathcal{L})}(m)(k_1) < |\mathfrak{A}'| \leq \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(k_1+1)$, resp. $\mathfrak{B}' \equiv_{m,\mathcal{L}} \mathfrak{B}$ and $\theta_{(\mathcal{S}_2,\mathcal{L})}(m)(k_2) < |\mathfrak{B}'| \leq \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(k_2+1)$. Let $\mathfrak{C}' = O(\mathfrak{A}',\mathfrak{B}')$.
2. Assume w.l.o.g. that $|\mathfrak{A}| \geq \theta_{(\mathcal{S}_1,\mathcal{L})}(m)(k_1+1)$ and $|\mathfrak{B}| < \theta_{(\mathcal{S}_2,\mathcal{L})}(m)(k_2+1)$. Then consider the structure $\mathfrak{A}'$ as described above, and let $\mathfrak{C}' = O(\mathfrak{A}',\mathfrak{B})$.

In either case, we observe that $\mathfrak{C}'$ is a substructure of $\mathfrak{C}$ in $\mathcal{S}$ such that $\mathfrak{C}' \equiv_{m,\mathcal{L}} \mathfrak{C}$ and $|\mathfrak{C}'| \in \langle i-1 \rangle_f$. Whereby, $\mathcal{S}$ is an $\mathcal{L}$-fractal, and $\theta_{(\mathcal{S},\mathcal{L})}$ witnesses the $\mathcal{L}$-fractal property of $\mathcal{S}$.