Contemporary Mathematics

Methods for Algorithmic Meta Theorems

Martin Grohe and Stephan Kreutzer

Abstract. Algorithmic meta-theorems state that certain families of algorithmic problems, usually defined in terms of logic, can be solved efficiently. This is a survey of algorithmic meta-theorems, highlighting the general methods available to prove such theorems rather than specific results.

1. Introduction

Faced with the seeming intractability of many common algorithmic problems, much work has been devoted to studying restricted classes of admissible inputs on which tractability results can be retained. A particularly rich source of structural properties which guarantee the existence of efficient algorithms for many problems on graphs comes from structural graph theory, especially graph minor theory. It has been found that most generally hard problems become tractable on graph classes of bounded tree-width and many remain tractable on planar graphs or graph classes excluding a fixed minor.

Besides many specific results giving algorithms for individual problems, of particular interest are results that establish tractability of a large class of problems on specific classes of instances. These results come in various flavours. Here we are mostly interested in results that take a descriptive approach, i.e. results that use a logic to describe algorithmic problems and then provide general tractability results for all problems definable in that logic on specific classes of inputs. Results of this form are usually referred to as *algorithmic meta-theorems*. The first explicit algorithmic meta-theorem was proved by Courcelle [**3**] establishing tractability of decision problems definable in monadic second-order logic (even with quantification over edge sets) on graph classes of bounded tree-width, followed by similar results for monadic second-order logic with only quantification over vertex sets on graph classes of bounded clique-width [**4**], for first-order logic on graph classes of bounded degree [**45**], on planar graphs and more generally graph classes of bounded local tree-width [**23**], on graph classes excluding a fixed minor [**20**], on graph classes locally excluding a minor [**6**] and graph classes of bounded local expansion [**13**].

The natural counterpart to any algorithmic meta-theorem establishing tractability for all problems definable in a given logic L on specific classes of structures are corresponding lower bounds, i.e. results establishing intractability results for L

1991 *Mathematics Subject Classification.* Primary 68Q19; Secondary 68Q25.

with respect to structural graph parameters. Ideally, one would aim for results of the form: all problems definable in L are tractable if a graph class $\mathcal{C}$ has a specific property $\mathcal{P}$, such as bounded tree-width, but if $\mathcal{C}$ does not have this property, then there are L definable properties that are hard.

Early results on lower bounds have either focused on graph classes with very strong closure properties such as being closed under minors [**37**], or on very specific graph classes such as the class of all cliques [**4**]. Recently, however, much more general lower bounds have been established giving much tighter bounds on the tractability of monadic second-order logic [**30, 34**] and first-order logic [**31**] with respect to structural parameters.

In this paper we give a survey of the most important methods used to obtain algorithmic meta-theorems. All these methods have a model-theoretic flavor. In the first part of the paper we focus on upper bounds, i.e. tractability results. Somewhat different to the existing surveys on algorithmic meta-theorems [**26, 31**], this survey is organised along the core methods used to establish the results, rather than the specific classes of graphs they refer to. We put an emphasis on the most recent results not yet covered in the earlier surveys. In particular, we devote the longest section of this article to the recent linear time algorithm for deciding first-order definable properties on graphs of bounded expansion [**13**], which was established by a completely new technique that we call the "Colouring Technique" here. Moreover, for the first time we also survey lower bounds for algorithmic meta theorems, most of which are very recent as well.

2. Preliminaries

We assume familiarity with basic concepts of logic and graph theory and refer to the textbooks [**14, 27, 10**] for background. Our notation is standard; in the following we review a few important points.

If M, N are two sets, we define $M \dot\cup N$ as the *disjoint union* of M and N, obtained by taking the union of M and a copy N' of N disjoint from M. We write $\mathbb{Z}$ for the set of integers and $\mathbb{N}$ for the set of non-negative integers.

All graphs in this article are finite, undirected and simple, i.e., have no multiple edges and no self loops. We denote the vertex set of a graph G by $V(G)$ and its edge set by $E(G)$. We usually denote an edge between vertices v and w as vw, i.e., without parenthesis. We use standard graph theoretic notions like *sub-graphs*, *paths* and *cycles*, *trees* and *forests*, *connectedness* and *connected components*, the *degree* of a vertex, etc, without further explanation. Occasionally, we also need to work with *directed graphs* (for short: *digraphs*). Here we also use standard terminology.

A graph G is a *minor* of a graph H (we write $G \preceq H$) if G is isomorphic to a graph obtained from a subgraph of H by contracting edges. (*Contracting* an edge means deleting the edge and identifying its endvertices.) A graph H is an *excluded minor* for a class $\mathcal{C}$ of graphs if H is not a minor of any graph in $\mathcal{C}$.

All structures in this article are finite and without functions. Hence a *signature* is a finite set of relation symbols and constant symbols. Each relation symbol $R \in \sigma$ is equipped with its *arity* $ar(R) \in \mathbb{N}$. Let σ be a signature. A σ-structure A is a tuple consisting of a finite set $V(A)$ of elements, the *universe*, for each relation symbol $R \in \sigma$ of arity r an r-ary relation $R(A) \subseteq V(A)^r$, and for each constant symbol $c \in \sigma$ a constant $c(A) \in V(A)$. (Hence if σ contains constant symbols then $V(A)$ must be nonempty.) A signature is *relational* if it contains no constant symbols, and a structure is *relational* if its signature is. A signature is *binary*, if the arity of all relation symbols in it is at most 2. The *order* $|A|$ of a σ-structure A is $|V(A)|$ and its *size* $||A||$ is $|\sigma| + |V(G)| + \sum_{R\in\sigma} |R(G)|$.[1] For σ-structures A, B, we write $A \cong B$ to denote that A and B are isomorphic.

We may view graphs as $\{E\}$-structures, where E is a binary relation symbol. The *Gaifman-graph* $G(A)$ of a σ-structure A is the graph with vertex set $V(A)$ and edge set $\{bc :$ there is an $R \in \sigma$ and a tuple $\overline{a} \in R(A)$ such that $b, c \in \overline{a}\}$. Here and elsewhere, for a tuple $\overline{a} = (a_1, \ldots, a_k)$ and an element b we write $b \in \overline{a}$ instead of $b \in \{a_1, \ldots, a_k\}$.

We denote the class of all structures by $\mathcal{S}$ and the class of all graphs by $\mathcal{G}$. If $\mathcal{C}$ is a class of graphs, we let $\mathcal{S}(\mathcal{C})$ be the class of all structures with Gaifman graph in $\mathcal{C}$. If $\mathcal{C}$ is a class of structures and σ a signature, we let $\mathcal{C}(\sigma)$ be the class of all σ-structures in $\mathcal{C}$.

Let σ be a relational signature and $A, B \in \mathcal{S}(\sigma)$. Then A is a *substructure* of B (we write $A \subseteq B$) if $V(A) \subseteq V(B)$ and $R(A) \subseteq R(B)$ for all $R \in \sigma$. If $A \subseteq B$ and $R(A) = R(B) \cap V(A)^{ar(R)}$ for all $R \in \sigma$ then A is an *induced* substructure of B. For a set $W \subseteq V(B)$, we let $B[W]$ be the induced substructure of B with universe W, and we let $B \setminus W := B[V(B) \setminus W]$.

Formulas of *first-order logic* FO are built from variables ranging over elements of the universe of a structure, atomic formulas $R(t_1, \ldots, t_k)$ and $t_1 = t_2$, where the t_i are *terms*, i.e., variables or constant symbols, the usual Boolean connectives $\wedge, \vee, \rightarrow, \neg$, and existential and universal quantification $\exists x$, $\forall x$, where x is a variable.

In *monadic second order logic* MSO we also have "set variables" ranging over sets of elements of the universe, new atomic formulas $X(t)$, where X is a set variable and t a term, and quantification over set variables. In the context of algorithmic meta-theorems, an extension MSO$_2$ of MSO is often considered. MSO$_2$ is a logic only defined on graphs, and in addition to variables ranging over sets of vertices it has also variables ranging over sets of edges of a graph. The generalisation of MSO$_2$ to arbitrary structures is known as *guarded second-order logic* GSO. It has variables ranging over relations of arbitrary arities, but for relations of arity greater than one only allows guarded quantification $\exists X \subseteq R$ and $\forall X \subseteq R$, where R is a relation symbol.

We write $\varphi(x_1, \ldots, x_k)$ to denote that the free variables of a formula (of some logic) are among $x_1, \ldots, x_k$, and for a structure A and elements $a_1, \ldots, a_k \in V(A)$, we write $A \models \varphi[a_1, \ldots, a_k]$ to denote that A satisfies φ if x_i is interpreted by a_i. Furthermore, we let

$$\varphi(A) = \{(a_1, \ldots, a_k) \mid A \models \varphi[a_1, \ldots, a_k]\}.$$

If φ is a *sentence*, i.e., a formula without free variables, we just write $A \models \varphi$ to denote that A satisfies φ.

3. Algorithmic Meta-Theorems and Model-Checking Problems

As described in the introduction, the algorithmic meta-theorems we are interested in here have the following general form:

Algorithmic Meta Theorem (Nonuniform Version). *Let* L *be a logic (typically* FO *or* MSO*), $\mathcal{C}$ a class of structures (most often a class of graphs), and $\mathcal{T}$ a*

[1]Up to constant factors, $||A||$ corresponds to the size of a representation of A in an appropriate model of computation, random access machines with a uniform cost measure (cf. [**19**]).

class of functions on the natural numbers (typically the class of all linear functions or the class of all polynomial functions). Then for all L*-definable properties* π *of structures in* $\mathcal{C}$*, there is a function* $t \in \mathcal{T}$ *and an algorithm that tests if a given structure* $A \in \mathcal{C}$ *has property* π *in time* $t(||A||)$*.*

Of course we may also restrict the algorithm's consumption of memory space or other resources, but most known meta-theorems are concerned with running time. (One notable exception is [**15**].) We note that in MSO we can define NP-complete properties of graphs, for example 3-colourability. Hence unless P = NP, there are MSO-definable properties of graphs for which no polynomial time algorithm exists. All FO-definable properties of graphs have a polynomial time algorithm, but the exponent of the running time of the algorithm will usually depend on the formula defining the property. There are generally believed assumptions from parameterized complexity theory (see [**12, 21**]) which imply that for every constant c there are FO-definable properties of graphs that cannot be decided by an $O(n^c)$-algorithm, where n is the size of the input graph.

Stated as above, meta-theorems are non-uniform in the sense that there is no direct connection between a property π and the corresponding algorithm. It would certainly be desirable to be able to construct the algorithm from an L-definition of the property. Fortunately, we usually obtain such uniform versions of our meta-theorems, which can be phrased in the following form:

Algorithmic Meta Theorem (Uniform Version). *Let* L *be a logic,* $\mathcal{C}$ *a class of structures, and* $\mathcal{T}$ *a class of functions on the natural numbers. Then there is an algorithm that, given an* L*-sentence* φ *and a structure* $A \in \mathcal{C}$*, decides whether* A *satisfies* φ*. Moreover, for every* L*-sentence* φ *there is a function* $t_\varphi \in \mathcal{T}$ *such that the running time of the algorithm on input* φ, A *is bounded by* $t_\varphi(||A||)$*.*

Hence in this uniform version, our meta-theorems are just statements about the complexity of model-checking problems. The *model-checking problem for the logic* L *on the class* $\mathcal{C}$ *of structures* is the following decision problem:

Given an L-sentence φ and a structure $A \in \mathcal{C}$, decide if A satisfies φ.

We denote this problem by MC(L, $\mathcal{C}$). It is well-known that both MC(FO, $\mathcal{G}$) and MC(MSO, $\mathcal{G}$) are PSPACE-complete [**49**]. Hence we cannot hope to obtain polynomial time algorithms. We say that MC(L, $\mathcal{C}$) is *fixed-parameter tractable* (for short: *fpt*) if it can be decided by an algorithm running in time

$$f(k) \cdot n^c \tag{3.1}$$

for some function f and some constant c. Here k denotes the length of the input formula φ and n the size of the input structure A. We say that MC(L, $\mathcal{C}$) is fpt by *linear time parameterized algorithms* if we can let $c = 1$ in (3.1). Now for $\mathcal{T}$ being the class of all linear functions, we can concisely phrase our algorithmic meta-theorem as follows:

Algorithmic Meta Theorem (Uniform Version for Linear Time). *Let* L *be a logic and* $\mathcal{C}$ *a class of structures. Then* MC(L, $\mathcal{C}$) *is fpt by linear time parameterized algorithms.*

This is the form in which we usually state our meta-theorems. Even though we usually think of the L-sentence φ as being fixed, we may wonder what the dependence of the running time of our fixed-parameter tractable model-checking algorithm on φ is, i.e., what we can say about the function f in (3.1). For all known meta-theorems f is easily seen to be computable. However, f usually grows very quickly. Even for very simple classes $\mathcal{C}$ (such as the class of all trees, which is contained in all classes that appear in the meta-theorems surveyed in this article), it has been shown [**24**] that, under generally believed complexity theoretic assumptions, all fpt algorithms for $\mathrm{MC}(\mathrm{FO}, \mathcal{C})$, and hence for $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$, have a non-elementary running time (i.e., f grows faster than any stack of exponentials of fixed-height).

Part I: Upper Bounds

In this first part of the paper we consider the most successful methods for establishing algorithmic meta-theorems.

4. The Automata Theoretic Method

The automata theoretic approach to algorithmic meta-theorems can best be explained with a familiar algorithmic problem, *regular expression pattern matching.* The goal is to decide whether a text matches a regular expression (or equivalently, has a sub-string matching a regular expression). One efficient way of doing this is to translate the regular expression into a deterministic finite automaton and then run the automaton on the text. The translation of the regular expression into the automaton may cause an exponential blow-up in size, but running the automaton on the text can be done in time linear in the length of the text, so this leads to an algorithm running in time of $O(2^k + n)$, where k is the length of the regular expression and n the length of the text. In practise, we usually match a short regular expression against a long text. Thus k is much smaller than n, and this algorithm, despite its exponential running time, may well be the best choice.

We can use the same method for MSO model-checking on words, suitably encoded as relational structures. By the Büchi-Elgot-Trakhtenbrot Theorem [**2, 16, 48**], we can translate every MSO-formula φ to a finite automaton A_φ that accepts precisely the words satisfying the formula. Hence to test if a word W satisfies φ we just need to run A_φ on W and see if it accepts. This leads to a linear time fpt algorithm for MSO model-checking on the class of words. The same method even works for trees. Since we are going to use this result later, let us state it more formally: with every finite alphabet Σ we associate a signature τ_Σ consisting of two binary relation symbols E_L and E_R and a unary relation symbol P_a for every $a \in \Sigma$. A *binary* Σ*-tree* is a τ_Σ-structure whose underlying graph is a tree in which E_L is the "left-child relation" and E_R is the "right-child relation" and in which every vertex belongs to exactly one P_a. Let $\mathcal{B}_\Sigma$ denote the class of binary Σ-trees.

Theorem 4.1 ([**11, 46**]). *For every finite alphabet Σ, the model-checking problem $\mathrm{MC}(\mathrm{MSO}, \mathcal{B}_\Sigma)$ is fpt by a linear-time parameterized algorithm.*

5. The Reduction Method

In this section, we will show how *logical reductions* can be used to transfer algorithmic meta-theorems between classes of structures. We start by reviewing *syntactic interpretations* or *transductions*, a well known tool from model theory. Recall that for a formula $\varphi(\overline{x})$ and a structure A, we denote by $\varphi(A)$ the set of all tuples $\overline{a}$ such that $A \models \varphi[\overline{a}]$.

Definition 5.1. *Let σ, τ be signatures and let* $\mathrm{L} \in \{\mathrm{MSO}, \mathrm{FO}\}$*. A (one-dimensional)* L-transduction *from τ to σ is a sequence*

$$\Theta := \big(\varphi_{valid}, \varphi_{univ}(x), (\varphi_R(\overline{x}))_{R\in\sigma}, (\varphi_c(x))_{c\in\sigma}\big)$$

of $\mathrm{L}[\tau]$ *formulas where for all relation symbols $R \in \sigma$, the number of free variables in φ_R is equal to the arity of R. Furthermore, for all τ-structures A such that $A \models \varphi_{valid}$ and all constant symbols $c \in \sigma$ there is exactly one element $a \in V(A)$ satisfying φ_c.*

If A is a τ-structure such that $A \models \varphi_{valid}$ we define $\Theta(A)$ as the σ-structure B with universe $V(B) := \varphi_{univ}(A)$, $R(B) := \varphi_R(A)$ for each $R \in \sigma$ and $c(B) := a$, where a is the uniquely defined element with $\{a\} = \varphi_c(A)$.

Finally, if $\mathcal{C}$ is a class of τ-structures we let $\Theta(\mathcal{C}) := \{\Theta(A) : A \in \mathcal{C}, A \models \varphi_{valid}\}$.

Every L-transduction from τ to σ naturally defines a translation of L-formulas from $\varphi \in \mathrm{L}[\sigma]$ to $\varphi^* := \Theta(\varphi) \in \mathrm{L}[\tau]$. Here, φ^* is obtained from φ by recursively replacing

- first-order quantifiers $\exists x\varphi$ by $\exists x(\varphi_{univ}(x) \wedge \varphi^*)$ and quantifiers $\forall x\varphi$ by $\forall x(\varphi_{univ}(x) \rightarrow \varphi^*)$,
- second-order quantifiers $\exists X\varphi$ and $\forall X\varphi$ by $\exists X\big(\forall y(Xy \rightarrow \varphi_{univ}(y)) \wedge \varphi^*\big)$ and $\forall X\big(\forall y(Xy \rightarrow \varphi_{univ}(y)) \rightarrow \varphi^*\big)$ respectively and
- atoms $R(\overline{x})$ by $\varphi_R(\overline{x})$ and
- every atom $P(\overline{u})$, where $\overline{u}$ is a sequence of terms containing a constant symbol $c \in \sigma$ and $P \in \sigma \cup \{=\}$ by the sub-formula $\exists c'(\varphi_c(c') \wedge P(\overline{u}'),$ where $\overline{u}'$ is obtained from $\overline{u}$ by replacing c by c' and c' is a new variable not occurring elsewhere in $\overline{u}$.

The following lemma is easily proved (see [**27**]).

Lemma 5.2. *Let Θ be an* MSO*-transduction from τ to σ. Then for all* $\mathrm{MSO}[\sigma]$*-formulas and all τ-structures $A \models \varphi_{valid}$*

$$A \models \Theta(\varphi) \quad \Longleftrightarrow \quad \Theta(A) \models \varphi.$$

The analogous statement holds for FO*-transductions.*

Transductions map τ-structures to σ-structures and also map formulas back the other way. This is not quite enough for our purposes as we need a mapping that takes both formulas and structures over σ to formulas and structures over τ.

Definition 5.3. *Let σ, τ be signatures and* $\mathrm{L} \in \{\mathrm{MSO}, \mathrm{FO}\}$*. Let $\mathcal{C}$ be a class of σ-structures and $\mathcal{D}$ be a class of τ-structures. An* L-reduction from $\mathcal{C}$ to $\mathcal{D}$ *is a pair $(\Theta, \mathcal{A})$, where Θ is an* L*-transduction from τ to σ and $\mathcal{A}$ is a polynomial time algorithm that, given a σ-structure $A \in \mathcal{C}$, computes a τ-structure $\mathcal{A}(A) \in \mathcal{D}$ with $\Theta(\mathcal{A}(A)) \cong A$.*

The next lemma is now immediate.

Lemma 5.4. *Let $(\Theta, \mathcal{A})$ be an* L*-reduction from a class $\mathcal{C}$ of σ-structures to a class $\mathcal{D}$. If* $\mathrm{MC}(\mathrm{L}, \mathcal{D})$ *is fpt then* $\mathrm{MC}(\mathrm{L}, \mathcal{C})$ *is fpt.*

Note that if the algorithm $\mathcal{A}$ in the reduction is a linear time algorithm then we can also transfer fixed-parameter tractability by linear time parameterized algorithms from $\mathrm{MC}(\mathrm{L}, \mathcal{D})$ to $\mathrm{MC}(\mathrm{L}, \mathcal{C})$. Of course, we can also apply the lemma to prove hardness of $\mathrm{MC}(\mathrm{L}, \mathcal{D})$ in cases where $\mathrm{MC}(\mathrm{L}, \mathcal{C})$ is known to be hard. In our context, L-reductions play a similar role as polynomial time many-one reductions in complexity theory. We start with two simple applications of the reduction technique.

Example 5.5. The first example is a reduction from graphs to their complements. Let $\Theta := (\varphi_{valid}, \varphi_{univ}, \varphi_E)$ be a transduction from σ_{graph} to σ_{graph} defined by $\varphi_{valid} = \varphi_{univ} := true$ and $\varphi_E(x, y) := \neg E(x, y)$. Then, for all graphs G, $\Theta(G)$ is the complement of G, i.e. the graph $\overline{G} := (V(G), \overline{E(G)})$.

Now if $\mathcal{A}$ is the algorithm which on input G outputs $\overline{G}$, then $(\Theta, \mathcal{A})$ is a first-order reduction reducing a pair (G, φ), where G is a graph and $\varphi \in \mathrm{FO}$, to $(\overline{G}, \Theta(\varphi))$ such that $G \models \varphi$ if, and only if, $\overline{G} \models \Theta(\varphi)$.

Hence, if $\mathcal{C}$ is the class of complements of binary trees then this reduction together with Theorem 4.1 implies that $\mathrm{MC}(\mathrm{FO}, \mathcal{C})$ is fpt. The same is true for MSO.

Obviously, the same reduction shows that if $\mathcal{D}$ is any class of graphs for which we will show $\mathrm{MC}(\mathrm{FO}, \mathcal{D})$ to be fpt in the remainder of this paper, then $\mathrm{MC}(\mathrm{FO}, \mathcal{C})$ is also fpt, where $\mathcal{C}$ is the class of graphs whose complements are in $\mathcal{D}$. ⊣

Example 5.6. In this example, we transfer the fixed-parameter tractability of MSO model-checking from binary to arbitrary Σ-trees. Let us fix a finite alphabet Σ, and let σ_Σ be the signature consisting of the binary relation symbols E and a unary relation symbol P_a for every $a \in \Sigma$. A *Σ-tree* is a τ_Σ-structure whose underlying graph is a directed tree with edges directed away from the root in which every vertex belongs to exactly one P_a.

With every Σ-tree T we associate a binary Σ-tree B_T as follows: we order the children of each node of T arbitrarily. Now each node has a "first child" and a "last child", and each child of a node except the last has a "next sibling". We let B_T be the binary Σ-tree with $V(B_T) = V(T)$ such that $E_L(B_T)$ is the "first child" relation of T and $E_R(B_T)$ is the "next sibling" relation. Clearly, there is a linear time algorithm $\mathcal{A}$ that computes B_T from T. (More precisely, the algorithm computes "some" B_T constructed in the way described, because the tree B_T depends on the choice of the ordering of children.)

Now we define an MSO-transduction $\Theta = (\varphi_{valid}, \varphi_{univ}, \varphi_E, (\varphi_{P_a})_{a\in\Sigma})$ from τ_Σ to σ_Σ such that for every Σ-tree T we have $\Theta(B_T) \cong T$. We let $\varphi_{valid} = \varphi_{univ}(x) := true$ and $\varphi_{P_a}(x) := P_a(x)$ for all $a \in \Sigma$. Moreover, we let

$$\begin{aligned}\varphi_E(x, y) := \forall X \Big[\Big(&\forall z (E_L(x, z) \to X(z)) \\ &\wedge \forall z \forall z' \big((X(z) \wedge E_R(z, z')) \to X(z') \big) \Big) \to X(y) \Big].\end{aligned}$$

We leave it to the reader to verify that in B_T this formula indeed defines the edge relation of T. ⊣

The rest of this section is devoted to a more elaborate application of the reduction technique: we will show that monadic second-order logic is fpt on all classes of structures of bounded tree-depth, a concept introduced in [**40**]. We first need some preparation.

In the following, we shall work with directed trees and forests. As before, all edges are directed away from the root(s). The *height* of a vertex v in a forest is its distance from the root of its tree. The *height* of the forest is the maximum of the heights of its vertices. A vertex v is an *ancestor* of a vertex w if there is a path from v to w. Furthermore, v is a *descendant* of w if w is an ancestor of v.

We define the *closure* of a forest F to be the (undirected) graph $clos(F)$ with vertex set $V(clos(F)) := V(F)$ and edge set

$$E(clos(F)) := \{vw \mid v \text{ is an ancestor of } w\}.$$

DEFINITION 5.7 ([**40**]). *A graph G has* tree-depth d *if it is a sub-graph of the closure of a rooted forest F of height d. We call F a* tree-depth decomposition *of G.*

It was proved in [**40**] that there is an algorithm which, given a graph G of tree-depth at most d, computes a tree-depth decomposition in time $f(d) \cdot |G|$, for some computable function d. For our purposes, an algorithm for computing approximate tree-depth decompositions will be enough. One such algorithm simply computes a *depth-first search forest* of its input graph. In [**40**] it was shown that a simple path of length 2^d has tree-depth exactly d. As the tree-depth of a sub-graph $H \subseteq G$ is at most the tree-depth of G, no graph of tree-depth d can contain a path of length greater than 2^d. This implies the following lemma.

LEMMA 5.8. *Let G be a graph of tree-depth d and let F be a forest obtained from a depth-first search (DFS) in G. Let F' be the closure of F. Then $G \subseteq F'$ and the depth of F is at most 2^d.*

We will now define an FO-reduction from the class $\mathcal{D}_d$ of graphs of tree-depth at most d to the class of Σ_d-trees, for a suitable alphabet Σ_d. For simplicity, we only present the reduction for connected graphs. We fix d and let $k = 2^d$ and $\Sigma_d = \{0,1\}^{\leq k}$, the set of all $\{0,1\}$-strings of length at most k.

We first associate a Σ_d-tree T_G with every connected graph $G \in \mathcal{D}_d$. Let T be a DFS-tree of G. By Lemma 5.8, the height of T is at most k, and $G \subseteq clos(T)$. We define a Σ_d-labelling of the vertices of T to obtain the desired Σ_d-tree T_G as follows: let $v \in V(T)$ be a vertex of height h, and let $u_0, \dots, u_{h-1}, v$ be the vertices on the path from the root to v in T. Then v is labelled by $i_0 \dots i_{h-1} \in \{0,1\}^h \subseteq \Sigma_d$, where $i_j = 1 \leftrightarrow u_j v \in E(G)$. It is not difficult to show that T_G can be computed from G by a linear time algorithm $\mathcal{A}$.

We leave it to the reader to define a first-order transduction Θ from σ_{Σ_k} to $\{E\}$ such that for every connected graph $G \in \mathcal{D}_d$ we have $\Theta(T_G) = G$. This yields the desired reduction.

By Example 5.6 and Theorem 4.1 we obtain the next corollary.

COROLLARY 5.9. MC(MSO, $\mathcal{D}_d$) *is fpt by a linear time parameterized algorithm.*

6. The Composition Method

The next method we consider is based on *composition theorems* for first-order logic and monadic second-order logic, which allow to infer the formulas satisfied by a structure composed of simpler pieces from these pieces. The best known such composition theorems are due to Feferman and Vaught [**18**].

Let L be either first-order or monadic second-order logic. Recall that the *quantifier rank* of an L-formula is the maximum number of nested quantifiers in the formula (counting both first-order and second-order quantifiers). Let A be a structure, $\overline{a} = (a_1, \dots, a_k) \in V(A)^k$ and $q \in \mathbb{N}$. Then the *q-type of $\overline{a}$ in A* is the set $\mathrm{tp}^A_{\mathrm{L},q}(\overline{a})$ of all L-formulas $\varphi(\overline{x})$ of quantifier-rank at most q such that $A \models \varphi[\overline{a}]$.

As such, the type of a tuple is an infinite class of formulas. However, we can syntactically normalise first-order and second-order formulas so that every formula can effectively be transformed into an equivalent normalised formula of the same quantifier-rank and furthermore for every quantifier-rank there are only finitely many pairwise non-equivalent normalised formulas. Hence, we can represent types by finite sets of normalised formulas. We will do so tacitly whenever we work with types in this paper. The following basic composition lemma can easily be proved using Ehrenfeucht-Fraïssé games (see [**36**] for a proof).

Lemma 6.1. *Let A, B be σ-structures and $\overline{a} \in V(A)^k, \overline{b} \in V(B)^\ell$, and $\overline{c} \in V(A \cap B)^m$ such that all elements of $V(A \cap B)$ appear in $\overline{c}$. Let $q \in \mathbb{N}$, and let $\mathrm{L} \in \{\mathrm{FO}, \mathrm{MSO}\}$.*

Then $\mathrm{tp}^{A\cup B}_{\mathrm{L},q}(\overline{a}\overline{b}\overline{c})$ is uniquely determined by $\mathrm{tp}^{A}_{\mathrm{L},q}(\overline{a}\overline{c})$ and $\mathrm{tp}^{B}_{\mathrm{L},q}(\overline{b}\overline{c})$. Furthermore, there is an algorithm that computes $\mathrm{tp}^{A\cup B}_{\mathrm{L},q}(\overline{a}\overline{b}\overline{c})$ from $\mathrm{tp}^{A}_{\mathrm{L},q}(\overline{a}\overline{c})$ and $\mathrm{tp}^{B}_{\mathrm{L},q}(\overline{b}\overline{c})$.

As an application and an illustration of the method, we sketch a proof of Courcelle's well-known meta-theorem for monadic second-order logic on graphs of bounded tree width. A *tree decomposition* of a graph G is a pair (T, β) where T is a tree and β a mapping that assigns a subset $\beta(t) \subseteq V(G)$ with every $t \in V(T)$, subject to the following conditions:

(1) For every vertex $v \in V(G)$ the set $\{t \in V(T) \mid v \in \beta(t)\}$ is nonempty and connected in T.
(2) For every edge $vw \in E(G)$ there is a $t \in V(T)$ such that $v, w \in \beta(t)$.

The *width* of a tree decomposition (T, β) is $\max\{|\beta(t)| : t \in V(T)\} - 1$, and the *tree width* of a graph G is the minimum of the widths of all tree decompositions of G. Intuitively, tree width may be viewed as a measure for the similarity of a graph with a tree. Bodlaender [**1**] proved that there is an algorithm that, given a graph G of tree width w, computes a tree decomposition of G of width w in time $2^{O(w^3)}|G|$. It is an easy exercise to show that every graph G of tree-depth at most h also has tree-width at most h.

Theorem 6.2 ([**3**]). *Let $\mathcal{C}$ be a class of graphs of bounded tree-width. Then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is fpt by linear time parameterized algorithms.*

Proof sketch. Let $G \in \mathcal{C}$ and $\varphi \in \mathrm{MSO}$ be given. Let w be the tree width of G (which is bounded by some constant), and let q be the quantifier rank of φ. We first use Bodlaender's algorithm to compute a tree decomposition (T, β) of G of width w. We fix a root r of T arbitrarily. For every $t \in V(T)$, we let T_t be the sub-tree of T rooted at t, and we let G_t be the induced subgraph of G with vertex set $\bigcup_{u \in V(T_t)} \beta(u)$. Moreover, we let $\overline{b}_t$ be a $(w+1)$-tuple of vertices that contains precisely the vertices in $\beta(t)$. (Without loss of generality we assume $\beta(t)$ to be nonempty.)

Now, beginning from the leaves, we inductively compute for each $t \in V(T)$ the type $\mathrm{tp}^{G_t}_{\mathrm{MSO},q}(\overline{b}_t)$. We can do this by brute force if t is a leaf and hence $|G_t| \leq w+1$, and we use Lemma 6.1 if t is an inner node.

Finally, we check whether $\varphi \in \mathrm{tp}^{G_r}_{\mathrm{MSO},q}(\overline{b}_r) = \mathrm{tp}^{G}_{\mathrm{MSO},q}(\overline{b}_r)$. □

Courcelle's theorem can easily be generalised from graphs to arbitrary structures, and it can be extended from monadic to guarded second-order logic (and thus to MSO_2 on graphs). An alternative proof of Courcelle's Theorem is based on Theorem 4.1 and the reduction method of Section 5.

By a similar application of the composition method it can be proved that $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is fpt for all classes $\mathcal{C}$ of graphs of bounded clique width (see [**4**]). Further applications of the composition method can be found in [**5, 36**].

Finally, let us mention an analogue of Courcelle's theorem for logarithmic space, recently proved in [**15**]: for every class $\mathcal{C}$ of graphs of bounded tree width, there is an algorithm for $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ that uses space $O(f(k) \cdot \log n)$, where f is a computable function and k, n denote the size of the input formula and structure, respectively, of the model-checking problem.

7. Locality based arguments

In Section 5 we have seen how logical reductions can be used to transfer tractability results from a one class of structures to another. In this section we will look at a tool that will allow us to transfer tractability results from a class $\mathcal{C}$ of a structures to the class of all structures that locally look like a structure from $\mathcal{C}$.

We start with a simple example to explain the basic idea. Recall that a *homomorphism* from a graph H to a graph G is a function $\pi : V(H) \to V(G)$ such that whenever $uv \in E(H)$ then $\pi(u)\pi(v) \in E(G)$. The *graph homomorphism problem* asks, given two graphs H and G, whether there is a homomorphism from H to G. The homomorphism problem can trivially be solved in time $O(|G|^{|H|} \cdot |H|^2)$. The question is if we can solve it in time $f(|H|)|G|^c$, for some computable function f and constant c. In general, this is not possible, but it becomes possible if the graph G is "locally simple".

To explain the idea, suppose first that H is a connected graph. Then if there is a homomorphism π from H to G then the distance between any two vertices in the image $\pi(H)$ is at most $|H| - 1$. To exploit this observation, we define the *k-neighbourhood* of a vertex v in a graph G to be the subgraph of G induced by the set of all vertices of distance at most k from v. Then to test if there is a homomorphism from a connected k-vertex graph H to a graph G, we test for all $v \in V(G)$ whether the $(k-1)$-neighbourhood of v contains a homomorphic image of H. Of course in general, this does not help much, but it does help if the $(k-1)$-neighbourhoods in G are structurally simpler than the whole graph G. For example, if the girth of G (that is, the length of the shortest cycle) is at least k, then the $(k-1)$-neighbourhood of every vertex is a tree, and instead of testing whether there is a homomorphism from H to arbitrary graph we only need to test whether there is a homomorphism from H to a family of trees (of depth at most $k-1$), and this is much easier than the general homomorphism problem. Or if the maximum degree of G is d, then the order of the $(k-1)$-neighbourhood of every vertex in G is less than $(d+1)^k$, and instead of testing whether there is a homomorphism from H to graph of arbitrary size we only need to test whether there is a homomorphism from H to a family of graphs of size less than $(d+1)^k$. To apply the same method if H is not connected, we just check for each connected component of H separately if there is a homomorphism to G.

We can apply the same idea to the model-checking problem for first-order logic, because by Gaifman's Locality Theorem, first-order logic is *local* in the following sense: if σ is a relational signature and A is a σ-structure, we define the distance $d^A(a, b)$ between any two vertices $a, b \in V(A)$ to be the length of the shortest path from a to b in the Gaifman-graph $G(A)$ of A.[2] We define the *r-neighbourhood* $N_r^A(a)$ of a vertex $a \in V(A)$ to be the induced substructure of A with universe $\{b \mid d^A(a,b) \leq r\}$. A first-order formula $\varphi(x)$ is *r-local* if for every structure A and all $a \in V(A)$

$$A \models \varphi[a] \qquad \text{iff} \qquad N_r^A(a) \models \varphi[a].$$

Hence, truth of an r-local formula at an element a only depends on its r-neighbourhood. A *basic local sentence* is a first-order sentence of the form

$$\exists x_1 \dots \exists x_k \Big(\bigwedge_{1 \leq i < j \leq k} \operatorname{dist}(x_i, x_j) > 2r \wedge \bigwedge_{i=1}^{k} \vartheta(x_i) \Big) \tag{7.1}$$

where $\vartheta(x)$ is r-local. Here $\operatorname{dist}(x,y) > 2r$ is a first-order formula stating that the distance between x and y is greater than $2r$.

Theorem 7.1 (Gaifman's Locality Theorem[**25**]). *Every first-order sentence is equivalent to a Boolean combination of basic local sentences. Furthermore, there is an algorithm that, given a first-order formula as input, computes an equivalent Boolean combination of basic local sentences.*

We can exploit Gaifman's Theorem to efficiently solve the model-checking problem for first-order logic in structures that are "locally simple" as follows: Given a structure A and a first-order sentence φ, we first compute a Boolean combination φ' of basic local sentences that is equivalent to φ. Then we check for each of the basic local sentences appearing in φ' whether they hold in A. We can easily combine the results to check whether the Boolean combination φ' holds. To check whether a basic local sentence of the form (7.1) holds in A, we first compute the set $T(\vartheta)$ of all $a \in V(A)$ such that $N_r^A(a) \models \varphi[a]$, or equivalently, $A \models \varphi[a]$. For this, we only need to look at the r-neighbourhoods of the elements of a, and as we assumed A to be "locally simple", we can do this efficiently. It remains to check whether $T(\vartheta)$ contains k vertices of pairwise distance greater than $2r$. It turns out that this can be reduced to a "local" problem as well, and as A is "locally simple", it can be solved efficiently.

This idea yields the following lemma, which captures the core of the locality method. We say that first-order model-checking is *locally fpt* on a class $\mathcal{C}$ of structures if there is an algorithm that, given a structure $A \in \mathcal{C}$, an element $a \in V(A)$, a sentence $\varphi \in \mathrm{FO}$, and an $r \in \mathbb{N}$, decides whether $N_r^A(a) \models \varphi$ in time $f(r, |\varphi|) \cdot |A|^{O(1)}$, for some computable function f.

Lemma 7.2 ([**23, 6**]). *Let $\mathcal{C}$ be a class of structures on which first-order model-checking is locally fpt. Then* $\mathrm{MC}(\mathrm{FO}, \mathcal{C})$ *is fpt.*

Maybe surprisingly, there are many natural classes of graphs on which first-order model-checking is locally fpt, among them planar graphs and graphs of bounded degree. The most important of these are the classes of *bounded local tree width.* A class $\mathcal{C}$ of graphs has this property if for every $r \in \mathbb{N}$ there is a $k \in \mathbb{N}$ such that for every $G \in \mathcal{C}$ and every $v \in V(G)$ we have $\operatorname{tw}(N_r^G(v)) \leq k$. Examples of classes of graphs of bounded local tree width are all classes of graphs that can be embedded in a fixed surface, all classes of bounded degree, and (trivially) all classes of bounded tree width. Bounded local tree width was first considered

[2]See Section 2 for a definition of Gaifman-graphs.

by Eppstein [**17**] in an algorithmic context (under the name "diameter tree width property").

In [**6**] Lemma 7.2 is applied in a context that goes beyond bounded local tree width to show that first-order model-checking is fpt on all classes of graphs locally excluding a minor.

8. Colouring and Quantifier-Elimination

In Section 7 we have seen a method for establishing tractability results based on structural properties of r-neighbourhoods in graphs. Another way of presenting the locality method is that we cover the graph by local neighbourhoods which have a simpler structure than the whole graph. More generally we could use other forms of covers, i.e. cover the graph by arbitrary induced sub-graphs whose structure is simple enough to allow tractable model-checking. The main difficulty is to infer truth of a formula in the whole graph from the truth of (possibly a set of) formulas in the individual sub-graphs used in the cover. In the case of neighbourhood covers, Gaifman's locality theorem provided the crucial step in the construction which allowed us to reduce the model-checking problem in the whole graph to model-checking in individual r-neighbourhoods.

In this section we present a similar method. Again the idea is that we cover the graph by induced sub-graphs. However this time r-neighbourhoods will not necessarily be contained in a single sub-graph. This will make combining model-checking results in individual sub-graphs to the complete graph much more complicated.

The method we present is based on vertex colourings of graphs. Basically, we will colour a graph with a certain number c of colours, where c will depend on the formula we want to check, such that for some $k < c$, the union of any k colours induces a sub-graph of simple structure.

This technique was first developed by DeVos et al. [**9**] for graph classes excluding a fixed minor. They showed that if $\mathcal{C}$ excludes a minor then there is a constant d such that any $G \in \mathcal{C}$ can be 2 coloured so that any colour class induces a graph of tree-width at most d. See also [**7, 8**] for generalisations and algorithmic versions of this result and various applications.

The technique was later generalised by Nešetřil and Ossona de Mendez to graph classes of bounded expansion [**38**] and to nowhere dense classes of graphs [**41**]. In this section we will show how this can be used to establish tractability results for first-order model-checking on graph classes of bounded expansion.

To formally define classes of bounded expansion we first need some preparation. Recall that a graph H is a *minor* of G if it can be obtained from a sub-graph $G' \subseteq G$ by contracting edges. An equivalent, sometimes more intuitive, characterisation of the minor relation can be obtained using the concept of *images*. An *image map* of H into G is a map μ mapping each $v \in V(H)$ to a tree $\mu(v) \subseteq G$ and each edge $e \in E(H)$ to an edge $\mu(e) \in E(G)$ such that if $u \neq v \in V(H)$ then $\mu(v) \cap \mu(u) = \varnothing$ and if $uv \in E(H)$ then $\mu(uv) = u'v'$ for some $u' \in V(\mu(u))$ and $v' \in V(\mu(v))$. The union $\bigcup_{v \in V(H)} \mu(v) \cup \bigcup_{e \in E(H)} \mu(e) \subseteq G$ is called the *image* of H in G. It is not difficult to see that $H \preccurlyeq G$ if, and only if, there is an image of H in G.

The *radius* of a graph is G is the least r such that there is a vertex $v \in V(G)$ with $G = N_r^G(v)$. For $r \geq 0$, a graph H is an *minor at depth* r of a graph G, denoted $H \preccurlyeq_r G$, if H has an image map μ in G where for all $v \in V(H)$, $\mu(v)$ is a tree of radius at most r.

Definition 8.1 (bounded expansion). *Let G be a graph. The* greatest reduced average density *of G with rank r is*

$$\nabla_r(G) := \max\Big\{\frac{|E(H)|}{|V(H)|} : H \preccurlyeq_r G\Big\}.$$

A class $\mathcal{D}$ of graphs has bounded expansion *if there is a computable*[3] *function $f : \mathbb{N} \to \mathbb{N}$ such that $\nabla_r(G) \leq f(r)$ for all $G \in \mathcal{D}$ and $r \geq 0$. Finally, a class $\mathcal{C}$ of σ-structures has* bounded expansion *if $\{G(A) : A \in \mathcal{C}\}$ has bounded expansion, where $G(A)$ denotes the Gaifman-graph of the structure A (see Section 2).*

As every graph of average degree at least $c \cdot k\sqrt{\log k}$, for some constant c, contains a k-clique as a minor [**28, 29, 47**], it follows that every class of graphs excluding a minor also has bounded expansion.

The next definition formally defines the concept of colourings such that any constant number of colour classes together induce a sub-graph of small tree-depth.

Definition 8.2. *Let σ be a signature. Let $\mathcal{C}$ be a class of σ-structures of bounded expansion and let $A \in \mathcal{C}$.*

(1) Let $\gamma : V(A) \to \Gamma$ be a vertex colouring of A. If $\overline{C} \in \Gamma^s$ is a tuple of colours, we write $A_{\overline{C}}$ for the sub-structure of A induced by the union $\{v \in V(A) : \gamma(v) \in \overline{C}\}$ of the colour classes in $\overline{C}$.

(2) For $k \geq 0$, a vertex-colouring $\gamma : V(A) \to \Gamma$ of A is a td-k-colouring *if $A_{\overline{C}}$ has tree-depth at most k, for all $\overline{C} \in \Gamma^k$.*

It was shown in [**38, 39**] that for graph classes of bounded expansion, td-k-colourings using a constant number of colours exist and can be computed efficiently.

Theorem 8.3 ([**38, 39**]). *If $\mathcal{C}$ is a class of σ-structures of bounded expansion then there are computable functions $f, N_{\mathcal{C}} : \mathbb{N} \to \mathbb{N}$ and an algorithm which, given $A \in \mathcal{C}$ and k, computes a td-k-colouring of A with at most $N_{\mathcal{C}}(k)$ colours in time $f(k) \cdot |A|$.*

To demonstrate the application of td-k-colourings for model-checking, we prove the following result that will be used later.

Theorem 8.4 ([**42**]). *Let σ be a signature and let $\mathcal{C}$ be a class of σ-structures of bounded expansion. There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that given an existential first-order formula $\varphi \in \mathrm{FO}[\sigma]$ and a σ-structure $A \in \mathcal{C}$, $A \models \varphi$ can be decided in time $f(|\varphi|) \cdot |A|$.*

Proof. W.l.o.g. we assume that φ is in prenex normal form, i.e. of the form

$$\varphi := \exists x_1 \ldots \exists x_q \vartheta,$$

with ϑ quantifier-free. Let q be the number of quantifiers in φ.

Using Theorem 8.3 we first compute a td-q-colouring $\gamma : V(A) \to \Gamma$ of A, where Γ is a set of $N_{\mathcal{C}}(q)$ colours, in time $f_1(q) \cdot |A|$, for some computable function f_1.

Clearly, $A \models \varphi$ if, and only if, there are vertices $a_1, \ldots, a_q \in V(A)$ such that $A \models \vartheta[\overline{a}]$ and therefore $A_{\gamma(a_1),\ldots,\gamma(a_q)} \models \varphi$. Hence, $A \models \varphi$ if, and only if, there is a tuple $\overline{C} \in \Gamma^q$ such that $A_{\overline{C}} \models \varphi$. Therefore, to check whether $A \models \varphi$ we go through all tuples $\overline{C} \in \Gamma^q$ and decide whether $A_{\overline{C}} \models \varphi$. As all $A_{\overline{C}}$ have tree-depth

[3]The original definition in [**38**] does not require f to be computable. This would imply that some of the following fpt-algorithms are non-uniform.

at most q, by Corollary 5.9, this check can be performed in time $f_2(q) \cdot |A|$, where f_2 is a computable function.

Hence, the complete algorithm runs in time $\big(f_1(|\varphi|) + N_{\mathcal{C}}(|\varphi|)^q \cdot f_2(|\varphi|)\big) \cdot |A|$. □

We are now ready to state the main result of this section. Our presentation follows [**33**].

THEOREM 8.5 ([**13**]). *Let σ be a relational signature and let $\mathcal{C}$ be a class of σ-structures of bounded expansion. Then* MC(FO, $\mathcal{C}$) *is fpt by linear time parameterized algorithms.*

We first give a high-level description of the proof. Given a structure $A \in \mathcal{C}$ and a formula φ with at most q quantifiers, we will define an equivalence relation of finite index on q-tuples of elements such that if $\overline{a}, \overline{b}$ fall into the same equivalence class then they satisfy the same formulas of quantifier-rank at most q. The equivalence class of a tuple $\overline{a}$ is called its *full type*. Suppose that for each tuple $\overline{b}$ of length at most q we can compute its equivalence class. Then, to decide whether $A \models \varphi$, we can use the naive evaluation algorithm, i.e. for each quantifier we test all possibilities. However, as equivalent tuples satisfy the same formulas, we only need to check one witness for each equivalence class and therefore we can implement the evaluation algorithm in constant time, depending only on the size of the formula. For this to work we need to compute the full types by a linear time fpt algorithm.

We proceed in stages. As a first step we define for each k-tuple $\overline{C}$ of colours the *local type of quantifier-rank* q of a tuple $\overline{a}$ of elements in $A_{\overline{C}}$. If two tuples have the same local type in some $A_{\overline{C}}$ then they satisfy the same first-order formulas in $A_{\overline{C}}$ up to quantifier-rank q.

The second step is the definition of the *global type* of a tuple $\overline{a}$, which is simply the collection of all local types of $\overline{a}$ in the individual sub-graphs $A_{\overline{C}}$, for all $\overline{C}$ of length at most k (see Definition 8.13). We will show that global types can be defined by existential first-order formulas.

Finally, we will use the global types as the basis for the definition of full types. A full type of a tuple $\overline{a}$ describes the complete quantifier-rank q first-order type of $\overline{a}$ and therefore determines which formulas of quantifier-rank at most q are true at $\overline{a}$. The main difficulty is to decide which full types are realised in the structure. For this we will show that each full type can be described by an existential first-order formula.

The existential formulas describing full types in a structure A will not be over the structure A itself, but over an expansion of A by the edges of tree-depth decompositions. We first introduce these expansions and then define the various types we are using.

Recall (from Section 6) that the first-order q-type $\mathrm{tp}^{A}_{\mathrm{FO},q}(a)$ of an element $a \in V(A)$ in a structure A is the class of all first-order formulas $\varphi(x) \in \mathrm{FO}[\sigma]$ of quantifier-rank at most q such that $A \models \varphi[v]$. As we are only dealing with first-order logic in this section, we drop the index FO from now on.

Notation. For the rest of this section we fix a signature σ and a class $\mathcal{C}$ of σ-structures of bounded expansion. Let $k, q \geq 0$ and let $c := (3k + q + 4) \cdot 2^{k+q+1}$. Let $A \in \mathcal{C}$ be a structure and $\gamma : V(A) \to \Gamma$ be a td-$(k + q)$-colouring of A, where Γ is a set of $N_{\mathcal{C}}(k + q)$ colours. As before, for each tuple $\overline{C} \in \Gamma^{k+q}$, let $A_{\overline{C}}$ be the sub-structure of A induced by the elements $\{v \in V(A) : \gamma(v) \in \overline{C}\}$. For each $\overline{C}$ we fix a depth-first search (DFS) forest $F_{\overline{C}}$ of $G(A_{\overline{C}})$ and add a self-loop to every root of a tree in $F_{\overline{C}}$.

Finally, we agree that for the rest of this section all formulas are "normalised" (see beginning of Section 6). In particular, this implies that we can test effectively whether a formula belongs to a given type. To increase readability, the formulas stated explicity in this section will not be normalised. However, they can easily be brought into normalised form as the normalisation process for first-order formulas is effective. □

Recall that, by Lemma 5.8, the closure of a DFS-forest $F_{\overline{C}}$ is a tree-depth decomposition of $A[\overline{C}]$ and as $A_{\overline{C}}$ has tree-depth at most $k+q$ the lemma implies that the height of $F_{\overline{C}}$ is at most 2^{k+q}.

Definition 8.6. *(1) For $\overline{C} \in \Gamma^{k+q}$ we define $(A_{\overline{C}}, F_{\overline{C}})$ as the $\sigma\dot{\cup}\{F_{\overline{C}}\}$-expansion of $A_{\overline{C}}$ with $F_{\overline{C}}((A_{\overline{C}}, F_{\overline{C}})) := E(F_{\overline{C}})$.*

(2) We define $\tau(\Gamma, \sigma, k+q) := \sigma \mathbin{\dot{\cup}} \{F_{\overline{C}}, T_{\overline{C},t} : \overline{C} \in \Gamma^{k+q}$ and $t(x)$ is a finite set of formulas $\varphi(x) \in \mathrm{FO}[\sigma\dot{\cup}\{F_{\overline{C}}\}]$ of quantifier-rank at most $c\}$.

(3) The $\tau(\Gamma, \sigma, k+q)$-structure $A(\gamma)$ is defined as the $\tau(\Gamma, \sigma, k+q)$-expansion of A with $F_{\overline{C}}(A(\gamma)) := E(F_{\overline{C}})$, and

$$T_{\overline{C},t}(A(\gamma)) := \{v \in V(A) : t = \mathrm{tp}_c^{(A_{\overline{C}}, F_{\overline{C}})}(v)\}.$$

Note that $A(\gamma)$ depends on the particular choice of $F_{\overline{C}}$ and is therefore not unique. But the precise choice will never matter and all results remain true independent of a particular choice of DFS-forest.

Essentially, to obtain $A(\gamma)$ we fix a tree-depth decomposition for each sub-structure induced by $k+q$ colours and add the edges of the decomposition to A, giving them a different edge colour $F_{\overline{C}}$ for any tuple $\overline{C} \in \Gamma^{k+q}$. Furthermore, for each $v \in V(A)$ and each sub-structure $(A_{\overline{C}}, F_{\overline{C}})$ induced by $k+q$ colours $\overline{C}$ which contains v we label v by its q-type in $(A_{\overline{C}}, F_{\overline{C}})$. The reason we work with DFS-forests rather than general tree-depth decompositions is that we can add the edges of a DFS-forest to A without introducing new edges in the Gaifman-graph. Hence, if $\mathcal{C}$ is a class of σ-structures of bounded expansion then the class $\{A(\gamma) : A \in \mathcal{C}$ and γ a td-$(k+q)$-colouring of $A\}$ also has bounded expansion for all choices of γ and DFS-forests.

8.1. Local types.

We show next that the formulas true at a given tuple $\overline{a}$ in a sub-structure $(A_{\overline{C}}, F_{\overline{C}})$ only depend on the formulas true at each individual element a_i and the relative position of the a_i within the tree-depth decomposition. By adding the edges of the tree-depth decomposition to the structure $A(\gamma)$, this relative position becomes first-order definable in $A(\gamma)$, a fact that will be used later in our model-checking algorithm.

Definition 8.7. *Let $\overline{C} \in \Gamma^{k+q}$ be a tuple of colours and let $x, y \in V(A_{\overline{C}})$ be two vertices contained in the same tree in $F_{\overline{C}}$.*

- *The* least common ancestor $\mathrm{lca}_{\overline{C}}(x, y)$ *of x and y in $F_{\overline{C}}$ is the element of $F_{\overline{C}}$ of maximal height that is an ancestor of both x and y.*
- *We define* $\mathrm{lch}_{\overline{C}}(x, y)$ *to be the height of* $\mathrm{lca}_{\overline{C}}(x, y)$ *in $F_{\overline{C}}$ and define* $\mathrm{lch}_{\overline{C}}(x, y) :=$ *∞ if x and y are not in the same component of $F_{\overline{C}}$.*

The following simple lemma shows that $\mathrm{lch}_{\overline{C}}$ and $\mathrm{lca}_{\overline{C}}$ are first-order definable in $(A_{\overline{C}}, F_{\overline{C}})$ for all $\overline{C} \in \Gamma^{k+q}$.

LEMMA 8.8. *For all $r \leq 2^{k+q}$ there is a first-order formula $\mathrm{lch}_r^{\overline{C}}(x,y) \in \mathrm{FO}[F_{\overline{C}}]$ of quantifier-rank at most $2^{k+q}+1$ such that for all $a, b \in V(A_{\overline{C}})$ we have*

$$A(\gamma) \models \mathrm{lch}_r^{\overline{C}}(a,b) \iff (A_{\overline{C}}, F_{\overline{C}}) \models \mathrm{lch}_r^{\overline{C}}(a,b) \iff \mathrm{lch}_{\overline{C}}(a,b) = r$$

The next lemma says that truth of a formula $\varphi(\overline{x})$ of quantifier-rank at most q at a tuple $\overline{a} := (a_1, \dots, a_k)$ only depends on the relative position of the a_i and the formulas of quantifier-rank at most $(k+q)2^{k+q+1}$ true at each a_i.

LEMMA 8.9. *Let $\overline{C} \in \Gamma^{k+q}$ and let $\varphi(x_1, \dots, x_k) \in \mathrm{FO}[\sigma \cup \{F_{\overline{C}}\}]$ be a formula of quantifier-rank at most q.*

If $u_1, \dots, u_k, v_1, \dots, v_k \in V(A_{\overline{C}})$ are such that for all $1 \leq i \leq k$ and $1 \leq i \leq j \leq 2^{k+q}$,

$$\mathrm{tp}_{(k+q)2^{k+q+1}}^{(A_{\overline{C}}, F_{\overline{C}})}(v_j) = \mathrm{tp}_{(k+q)2^{k+q+1}}^{(A_{\overline{C}}, F_{\overline{C}})}(u_j) \textit{ and } \mathrm{lch}(v_i, v_j) = \mathrm{lch}(u_i, u_j)$$

then $(A_{\overline{C}}, F_{\overline{C}}) \models \varphi(v_1, \dots, v_k)$ if, and only if, $(A_{\overline{C}}, F_{\overline{C}}) \models \varphi(u_1, \dots, u_k)$.

We are now ready to define the first equivalence relation on tuples of vertices, the *local type* of a tuple.

DEFINITION 8.10 (Local Types).
(1) For all $\overline{C} \in \Gamma^{k+q}$ we define the set $\mathrm{Loc}(\overline{C}, \sigma, k, q)$ of local types *as the set of all tuples*

$$\big(t_1, \dots, t_k, (r_{i,j})_{1 \leq i < j \leq k}\big),$$

where t_i is a finite set of formulas $\varphi(x) \in \mathrm{FO}[\sigma \dot\cup \{F_{\overline{C}}\}]$ of quantifier-rank at most c and $r_{i,j} \leq 2^{k+q}$ for all i, j.

(2) For $\overline{C} \in \Gamma^{k+q}$ and $\overline{a} := a_1, \dots, a_k \in V(A_{\overline{C}})$ we define the local type *$loc_q(\overline{a}; \overline{C}) \in \mathrm{Loc}(\overline{C}, k, q)$ as*

$$\big(\mathrm{tp}_c^{(A_{\overline{C}}, F_{\overline{C}})}(a_1), \dots, \mathrm{tp}_c^{(A_{\overline{C}}, F_{\overline{C}})}(a_k), (\mathrm{lch}_{\overline{C}}(a_i, a_j))_{1 \leq i < j \leq k}\big).$$

We will prove next that the local type $loc_q(\overline{a}; \overline{C}) := (t_1, \dots, t_k, (r_{i,j})_{1 \leq i < j \leq k})$ of a tuple $\overline{a}$ of vertices completely describes the formulas of quantifier-depth at most q which are true at $\overline{a}$ within the sub-structure $(A_{\overline{C}}, F_{\overline{C}})$. Note that we require the t_i to be quantifier-rank c-types of a_i even though we are only interested in formulas $\varphi(x_1, \dots, x_k)$ of quantifier-rank q. The reason for this will become clear in the following lemma.

LEMMA 8.11. *Let $l := (t_1, \dots, t_k, (r_{i,j})_{1 \leq i < j \leq k}) \in \mathrm{Loc}(\overline{C}, \sigma, k, q)$ be a local type. Then for all formulas $\varphi(\overline{x})$ with quantifier-rank at most q and all k-tuples $\overline{a} \in V(G)^k$ with $loc_q(\overline{a}, \overline{C}) = l$, $(A_{\overline{C}}, F_{\overline{C}}) \models \varphi[\overline{a}]$ if, and only if, if t_1 contains the formula*

$$\varphi^*(x_1) := \exists x_2 \dots \exists x_k \begin{array}{l} \bigwedge_{1 \leq i < j \leq k} \mathrm{lch}_{r_{i,j}}^{\overline{C}}(x_i, x_j) \ \wedge \\ \bigwedge_{1 \leq i \leq k} \bigvee_{t \in \mathrm{Loc}(\overline{C}, \sigma, k, q) : t_i \cap \mathcal{T}_{(k+q) \cdot 2^{k+q+1}} \subseteq t} \mathcal{T}_{\overline{C}, t}(x_i) \ \wedge \\ \varphi(x_1, \dots, x_k), \end{array}$$

where $\mathcal{T}_{(k+q) \cdot 2^{k+q+1}}$ is the finite set of all formulas in $\mathrm{FO}[\sigma \dot\cup \{F_{\overline{C}}\}]$ of quantifier-rank at most $(k+q) \cdot 2^{k+q+1}$.

Proof. Recall that the height of $F_{\overline{C}}$ is at most 2^{k+q}. Hence, by Lemma 8.8, the quantifier-rank of the formula φ^* is at most c.

Suppose $(A_{\overline{C}}, F_{\overline{C}}) \models \varphi[\overline{a}]$. Choosing $a_1, \dots, a_k$ as witnesses for $x_1, \dots, x_k$ it is obvious that $(A_{\overline{C}}, F_{\overline{C}}) \models \Big(\bigwedge_{1 \leq i < j \leq k} \mathrm{lch}_{r_{i,j}}(x_i, x_j) \wedge \bigwedge_{1 \leq i \leq k} T_{\overline{C}, t_i}(x_i) \Big)[\overline{a}]$. Hence, $\varphi^*(x_1)$ is contained in t_1.

Conversely, suppose that $\varphi^*(x_1)$ is contained in t_1 and hence $(A_{\overline{C}}, F_{\overline{C}}) \models \varphi^*[a_1]$. Hence, there are $b_2, \dots, b_k \in V(A_{\overline{C}})$ such that $\mathrm{lch}_{\overline{C}}(b_i, b_j) = r_{i,j}$, for all $1 \leq i < j \leq k$, where we set $b_1 := a_1$ to simplify notation, and further $\mathrm{tp}^{(A_{\overline{C}}, F_{\overline{C}})}_{(k+q)\cdot 2^{k+q+1}}(b_i) = \mathrm{tp}^{(A_{\overline{C}}, F_{\overline{C}})}_{(k+q)\cdot 2^{k+q+1}}(a_i)$, for all $1 \leq i \leq k$. Hence, by Lemma 8.9, $\overline{a}$ and $\overline{b}$ satisfy the same formulas of quantifier-rank at most q in $(A_{\overline{C}}, F_{\overline{C}})$ and therefore $(A_{\overline{C}}, F_{\overline{C}}) \models \varphi[\overline{a}]$. □

Recall that we are only working with normalised formulas in this section. However, the normalisation process for first-order formulas is effective and hence a normalised version of the formula φ^* can be computed effectively from the formula φ. Hence, the lemma implies that whether a tuple $\overline{a}$ with local type l satisfies a formula $\varphi(\overline{x})$ within some $(A_{\overline{C}}, F_{\overline{C}})$ can be read off directly from the local type l independent of the actual tuple $\overline{a}$. This motivates the following definition.

Definition 8.12. *A local type l defined as $l := (t_1, \dots, t_k, (r_{i,j})_{1 \leq i < j \leq k}) \in \mathrm{Loc}(\overline{C}, \sigma, k, q)$ entails a formula $\varphi(x_1, \dots, x_k)$ of quantifier-rank at most q, denoted $l \models \varphi$, if t_1 contains the formula $\varphi^*(x_1)$ defined in Lemma 8.11.*

8.2. Global types. As the second step towards defining the full type of a tuple $\overline{a}$ we now define the *global type* of $\overline{a}$, which is the collection of their local types over all combinations of colours.

Definition 8.13. *(1) We define* $\mathrm{Glob}(\Gamma, \sigma, k, q) := \{ (l_{\overline{C}})_{\overline{C} \in \Gamma^{k+q}} : l_{\overline{C}} \in \mathrm{Loc}(\overline{C}, \sigma, k, q) \}$.

(2) For $\overline{a} \in V(G)$ we define the global type *of $\overline{a}$ as*

$$glob_q(\overline{a}, \Gamma) := \big(loc_q(\overline{a}, \overline{C}) \big)_{\overline{C} \in \Gamma^{k+q}} \in \mathrm{Glob}(\Gamma, \sigma, k, q).$$

We now extend Lemma 8.11 to tuples having the same global type in G. However, this only applies to existential formulas and can be shown to be false for formulas with quantifier alternation.

Lemma 8.14. *If $\overline{a} := a_1, \dots, a_k, \overline{b} := b_1, \dots, b_k \in V(G)$ are tuples such that $glob_q(\overline{a}) = glob_q(\overline{b})$, then $\overline{a}$ and $\overline{b}$ satisfy in A the same existential formulas $\varphi \in$* FO$[\sigma]$ *with at most q quantifiers.*

More precisely, $A \models \varphi[\overline{a}]$ if, and only if, $glob_q(\overline{a})$ contains a local type l which entails φ.

Proof. Let $\varphi(x_1, \dots, x_k) \in \mathrm{FO}[\sigma]$ be an existential first-order formula with at most q quantifiers. W.l.o.g. we assume that φ is in prenex normal form, i.e. $\varphi := \exists z_1 \dots z_q \vartheta(\overline{x}, \overline{z})$, where ϑ is quantifier-free.

Suppose $A \models \varphi[\overline{a}]$. Let $u_1, \dots, u_q$ be witnesses for the existential quantifiers in φ, i.e. $A \models \vartheta[\overline{a}, \overline{u}]$, and let $\overline{C} := (\gamma(a_1), \dots, \gamma(a_k), \gamma(u_1), \dots, \gamma(u_q))$. Then, $A_{\overline{C}} \models \varphi[\overline{a}]$ and therefore $loc_q(\overline{a}, \overline{C})$ entails φ. As $\overline{b}$ has the same global type as $\overline{a}$ it also has the same local types, i.e. $loc_q(\overline{b}, \overline{C}) = loc_q[\overline{a}, \overline{C}]$ and therefore $A_{\overline{C}} \models \varphi(\overline{b})$. As the argument is symmetric, this concludes the proof. □

Again we define entailment between types and formulas.

DEFINITION 8.15. *Let $l := \big(l_{\overline{C}}\big)_{\overline{C}\in\Gamma^{k+q}} \in \mathrm{Glob}(\Gamma,\sigma,k,q)$ and let $\varphi(\overline{x}) \in \mathrm{FO}[\sigma]$ be an existential formula with at most q quantifiers and k free variables $x_1,\dots,x_k$. The type l* entails *φ, denoted $l \models \varphi$, if there is $\overline{C} \in \Gamma^{k+q}$ such that $l_{\overline{C}}$ entails φ.*

LEMMA 8.16. *For each $l \in \mathrm{Glob}(\Gamma,\sigma,k,q)$ there is an existential first-order formula $\varphi_l(\overline{x})$ such that for all $\overline{a} \in V(A)^k$,*

$$glob_q(\overline{a},\Gamma) = l \quad \textit{if, and only if,} \quad A(\gamma) \models \varphi_l[\overline{a}].$$

Furthermore, the formula depends only on Γ, σ, k and q but not on a specific colouring or structure.

Proof. Suppose $l := (l_{\overline{C}})_{\overline{C}\in\Gamma^{k+q}}$, where $l_{\overline{C}} := (t_1,\dots,t_k,(r_{i,j})_{1\le i<j\le k})$ are local types. For each $l_{\overline{C}}$ define

$$\varphi_{l_{\overline{C}}}(\overline{x}) := \bigwedge_{i=1}^{k} x_i \in P_{\overline{C},t_i} \wedge \bigwedge_{1\le i<j\le k} \mathrm{lch}^{\overline{C}}_{r_{i,j}}(x_i,x_j).$$

Then, $A(\gamma) \models \varphi_{l_{\overline{C}}}[\overline{a}]$ if, and only if, $loc_q(\overline{a};\overline{C}) = l_{\overline{C}}$. Hence.

$$\varphi_l(\overline{x}) := \bigwedge_{\overline{C}\in\Gamma^{k+q}} \varphi_{l_{\overline{C}}}(\overline{x})$$

says that the global type of $\overline{x}$ is l. □

8.3. Full Types. Finally, we give the definition of full types, the main equivalence relation between tuples used in our algorithm. We will define the full type $\mathrm{ft}^q_i(\overline{a})$ of an i-tuple $\overline{a} \in V(A)^i$ such that if $\overline{a}$ and $\overline{b}$ have the same full type they satisfy the same formulas of quantifier-rank at most $q-i$.

DEFINITION 8.17. *For $0 \le i \le q$ we define the set $\mathfrak{F}^q_i$ of full types of i-tuples and the full type $\mathrm{ft}^q_i(\overline{a})$ of $\overline{a} := a_1\dots a_i \in V(A)^i$ inductively as follows.*

(1) For $i = q$ we set $\mathfrak{F}^q_q := \mathrm{Glob}_{\mathcal{C}}(\Gamma,\sigma,q,0)$ and for $a_1,\dots,a_q \in V(A)$ we define $\mathrm{ft}^q_q(\overline{a}) := glob_0(\overline{a},\Gamma)$.

(2) For $i < q$ we define $\mathfrak{F}^q_i := \{\Phi : \Phi \subseteq \mathfrak{F}^q_{i+1}\}$ and for $\overline{a} := a_1,\dots,a_i$

$$\mathrm{ft}^q_i(\overline{a}) := \{\mathrm{ft}^q_{i+1}(\overline{a},a_{i+1}) : a_{i+1} \in V(A)\}.$$

A full type $t \in \mathfrak{F}^q_i$ is realised in A and γ *if there is $\overline{a} \in V(A)^i$ such that $t = \mathrm{ft}^q_i(\overline{a})$. We define $\mathfrak{R}^q_i(A,\gamma) \subseteq \mathfrak{F}^q_i$ as the set of types realised in A and γ.*

Note that the cardinality of $\mathfrak{F}^q_i$ only depends on q and $\mathcal{C}$. A straight forward Ehrenfeucht-Fraïssé-game argument establishes the following lemma.

LEMMA 8.18. *If $\overline{a},\overline{b} \in V(G)^i$ are such that $\mathrm{ft}^q_i(\overline{a}) = \mathrm{ft}^q_i(\overline{b})$ then $\overline{a}$ and $\overline{b}$ satisfy the same formulas of quantifier-rank at most $q-i$ in A.*

We show next that the full type of a tuple can be described by an existential first-order formula. As a consequence, we can check in linear time whether a full type is realised. For this, we first establish two lemmas which show that we can express Boolean combinations of existential formulas in a structure $A \in \mathcal{C}$ by an existential formula. However, this formula will not be over A but over an expansion $A(\gamma)$ for a suitable td-l-colouring $\gamma : V(A) \to \Gamma$, for some l. The next lemmas therefore no longer refer to the structure $A \in \mathcal{C}$ and colouring γ fixed at the beginning and we therefore state them in full generality.

LEMMA 8.19. *Let $k, q \geq 0$. Let $\mathcal{D}$ be a class of σ'-structures of bounded expansion and let $\varphi(\overline{x}) \in \mathrm{FO}[\sigma']$ be an existential formula with q quantifiers and k free variables $\overline{x} := x_1, \dots, x_k$. Let Γ be a set of $N_{\mathcal{D}}(k+q)$ colours.*

There is an existential formula $\overline{\varphi}(\overline{x}) \in \mathrm{FO}[\tau(\Gamma, \sigma', q+k)]$ such that for all $A \in \mathcal{D}$ and all td-$(q+k)$-colourings $\gamma : V(A) \to \Gamma$ and all $\overline{a} \in V(A)^k$

$$A \not\models \varphi[\overline{a}] \quad \textit{if, and only if,} \quad A(\gamma) \models \overline{\varphi}[\overline{a}].$$

Proof. W.l.o.g. we can assume that φ is in prenex normal form, i.e. of the form $\varphi := \exists \overline{y} \vartheta$, where ϑ is quantifier-free.

By Lemma 8.14, $A \models \varphi[\overline{a}]$ if, and only if, $glob_q(\overline{a}, \Gamma) \models \varphi$. It follows that $\overline{a}$ does not satisfy φ in A if $glob_q(\overline{a}, \Gamma) \not\models \varphi$.

As, by Lemma 8.16, global types l can be expressed by existential formulas φ_l, we can express that $A \not\models \varphi(\overline{a})$ by the existential $\mathrm{FO}[\tau(\Gamma, \sigma', q+k)]$-formula

$$\overline{\varphi}(\overline{x}) := \bigvee_{l \in \mathrm{Glob}(\Gamma, \sigma', k, q), l \not\models \varphi(\overline{x})} \varphi_l(\overline{x}).$$

□

LEMMA 8.20. *Let $k, q \geq 0$. Let $\mathcal{D}$ be a class of σ'-structures of bounded expansion and let $\varphi_1(\overline{x}), \dots, \varphi_n(\overline{x}) \in \mathrm{FO}[\sigma']$ be existential formulas with k free variables each.*

Then there is a $q \geq 0$ and a set Γ of $N_{\mathcal{C}}(k+q)$ colours and for each $I \subseteq \{1, \dots, n\}$ an existential formula $\varphi_I \in \mathrm{FO}[\tau(\Gamma, \sigma', k+q)]]$ such that for all $A \in \mathcal{D}$ and all td-$(q+k)$-colourings $\gamma : V(A) \to \Gamma$ and all $\overline{a} \in V(A)^k$,

$$A \models \psi_I[\overline{a}] \quad \textit{if, and only if, } A(\gamma) \models \varphi_I[\overline{a}],$$

where $\psi_I := \bigwedge_{i \in I} \exists x \varphi_i \wedge \bigwedge_{i \notin I} \neg \exists x \varphi_i(x)$.

Proof. To define an existential formula φ_I equivalent to ψ_I we have to replace the $\neg \exists \varphi_i(x)$ parts by existential statements. Let q' be the maximum number of quantifiers in any φ_i, $1 \leq i \leq n$ and let $q := q' + 1$. Let Γ be a set of $N_{\mathcal{C}}(k+q)$ colours.

As $\exists x \varphi_i$ is an existential formula, Lemma 8.19 implies that there is an existential $\mathrm{FO}[\tau(\Gamma, \sigma', k+q)]$-formula $\overline{\varphi}$ such that for all td-$(k+q)$-colourings $\gamma : V(A) \to \Gamma$, $A \not\models \exists x \varphi_i(\overline{a})$ if, and only if, $A(\gamma) \models \overline{\varphi}_i(\overline{a})$.

Hence, for all $I \subseteq \{1, \dots, n\}$

$$A \models \psi(\overline{a}) \quad \text{if, and only if,} \quad A(\gamma) \models \bigwedge_{i \in I} \exists x \varphi_i \wedge \bigwedge_{i \notin I} \overline{\varphi}_i$$

□

The previous two lemmas immediately imply the following.

LEMMA 8.21. *Let $\mathcal{D}$ be a class of σ'-structures of bounded expansion. Let $q \geq 0$. Let $A \in \mathcal{D}$ and $\gamma : V(A) \to \Gamma$ be a td-q-colouring.*

There is $r := r(q, \sigma', \mathcal{D}) \in \mathbb{N}$ such that for all $1 \leq i \leq q$ and all $l \in \mathfrak{F}_i^q$ there are existential first-order formulas $\varphi_l(x_1, \dots, x_i), \varphi_l(x_1, \dots, x_i), \varphi_l^e, \varphi_l^{\neg e} \in \mathrm{FO}[\tau(\Gamma', \sigma, r)]$, where Γ' is a set of $N_{\mathcal{C}}(r)$ colours disjoint from Γ, such that for every $A \in \mathcal{D}$,

$\overline{a} \in V(A)^i$ *and td-r-colouring* $\gamma' : V(A) \to \Gamma'$

$$\begin{array}{rll} A(\gamma') \models \varphi_l[\overline{a}] & \textit{if, and only if,} & \mathrm{ft}_i^q(\overline{a}) = l. \\ A(\gamma') \models \varphi_l^e & \textit{if, and only if,} & \textit{the type } l \textit{ is realised in } A(\gamma) \\ A(\gamma') \models \varphi_l^{\neg}[\overline{a}] & \textit{if, and only if,} & \mathrm{ft}_i^q(\overline{a}) \neq l. \\ A(\gamma') \models \varphi_l^{\neg e} & \textit{if, and only if,} & \textit{the type } l \textit{ is not realised in } A(\gamma). \end{array}$$

Proof. For $l \in \mathfrak{F}_q^q$ the existence of φ_l was proved in Lemma 8.16. Then $\varphi_l^e := \exists \overline{x} \varphi_l$. Furthermore, $\varphi_l^{\neg}$ and $\varphi_l^{\neg e}$ can be obtained from φ_l, φ_l^e by Lemma 8.19.

For the induction step, let $l \in \mathfrak{F}_i^q$ for some $i < q$. Then $l \subseteq \mathfrak{F}_{i+1}^q$ is a set of types $t \in \mathfrak{F}_{i+1}^q$ which, by induction hypothesis, can all be defined by existential formulas. Hence, $\varphi_l' := \bigwedge_{t \in l} \varphi_t \wedge \bigwedge_{t \notin l} \neg \varphi_t$ defines l and, by Lemma 8.20, can equivalently be written as an existential formula. $\varphi_l^e, \varphi_l^{\neg}, \varphi_l^{\neg e}$ can be defined as before.

Note that each step increases the signature and number of colours so that we finally obtain r and τ as required. □

As a consequence of the previous lemma we get that the set of types realised in a given structure can be computed in parameterized-linear time.

Corollary 8.22. *Let $\mathcal{C}$ be a class of bounded expansion. There is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that on input $A \in \mathcal{C}$, $q \geq 0$ and a td-q-colouring $\gamma : V(A) \to \Gamma$ the set $\mathfrak{R}_i^q$ can be computed in time $f(q) \cdot |G|$, for all $1 \leq i \leq q$.*

8.4. Model-checking in classes of bounded expansion. We are now going to describe our model-checking algorithm for classes of bounded expansion. Let $\mathcal{C}$ be a class of σ-structures of bounded expansion and $A \in \mathcal{C}$. Let $\varphi \in \mathrm{FO}[\sigma]$ be a formula with at most q quantifiers. W.l.o.g. we assume that φ is in prenex normal form and of the form $\varphi := \exists x_1 Q_2 x_2 \dots Q_q x_q \vartheta(x_1, \dots, x_q)$ with ϑ quantifier free and $Q_i \in \{\exists, \forall\}$. For $i \geq 1$ we define $\varphi_i(x_1, \dots, x_i) := Q_{i+1} x_{i+1} \dots Q_q x_q \vartheta$.

We can now check whether $A \models \varphi$ as follows. First, we compute a td-q-colouring $\gamma : V(A) \to \Gamma$ of A. By Corollary 8.22 there is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that the sets $\mathfrak{R}_i^q$, for $i \leq q$, can be computed in time $f(q) \cdot |A|$.

For each $t \in \mathfrak{R}_0^q$ we can now simply test whether $t \models \varphi$ and return true if such a type exists.

It is easily seen that the algorithm is correct. Furthermore, its running time only depends on the size of φ and the size of $\bigcup_{1 \leq i \leq q} \mathfrak{R}_{q-i}^i$ which again depends only on φ and σ.

Hence, by Corollary 8.22 and Theorem 8.3 the algorithm runs in time $f(|\varphi|) \cdot |G|$, for some computable function $f : \mathbb{N} \to \mathbb{N}$. This concludes the proof of Theorem 8.5.

Part II: Lower Bounds

In the previous part we have presented a range of tools for establishing tractability results for logics on specific classes of graphs. In this section we consider the natural counterparts to these results, namely lower bounds establishing limits beyond which the tractability results cannot be extended. Ideally, we aim for logics L such as FO, MSO_1 or MSO_2 for a structural property P_{L} such that model-checking for L is tractable on a class of structures if, and only if, it has the property P_{L}. As the general model-checking problem for $\mathrm{FO}, \mathrm{MSO}_1, \mathrm{MSO}_2$ is PSPACE-complete, any proof that model-checking for any of the logics is not fpt on a class $\mathcal{C}$ would separate

PTIME from PSPACE. We can therefore only hope to find such a property subject to assumptions from complexity theory and possibly subject to further restrictions.

In this section we first review recent lower bounds for monadic second-order logic with edge set quantification (MSO_2) and then comment briefly on lower bounds for FO and MSO_1.

9. Lower Bounds for MSO with Edge Set Quantification

In this section we review the known lower bounds for monadic second-order logic MSO_2. To make the results as strong as possible, we will concentrate on simple undirected graphs.

Recall that by Courcelle's theorem (Theorem 6.2), MSO_2 model-checking is fixed-parameter tractable on any class of structures of bounded tree-width. The aim of this section is to establish intractability results for classes of graphs of unbounded tree-width. As explained above, the lower bounds reported below are conditional on some complexity theoretical assumptions. Consequently, the results usually are proved by a reduction from some NP-hard problems.

At the core of all results reported below is the observation that the run of a Turing machine M on some input $w \in \{0,1\}^*$ can be simulated by an MSO_2 formula on a suitable sub-graph of a large enough grid. Here, the $(n \times m)$-grid is the graph $G_{n,m}$ with vertex set $\{(i,j) : 1 \leq i \leq n, 1 \leq j \leq m\}$ and edge set $\{\big((i,j),(i',j')\big) : |i-i'| + |j-j'| = 1\}$. Essentially, the grid provides the drawing board on which the time space diagram of a run of M on w can be guessed using set quantification. This yields the following result which is part of the folklore (see [**32**] for an exposition).

THEOREM 9.1. *Let $\mathcal{G}^* := \{H \subseteq G_{n\times n} : n > 0\}$ be the class of sub-graphs of grids. If* PTIME $\neq$ NP *then* $\mathrm{MC}(\mathrm{MSO}_1, \mathcal{G}^*)$ *is not fpt.*

We can use the result to obtain the following lower bound for MSO_1 on graph classes closed under taking minors, first obtained by Makowsky and Mariño.

THEOREM 9.2 ([**37**]). *Let $\mathcal{C}$ be a class of graphs closed under taking minors. If $\mathcal{C}$ has unbounded tree-width then* $\mathrm{MC}(\mathrm{MSO}_1, \mathcal{C})$ *is not fpt unless* PTIME = NP. *The same is true if $\mathcal{C}$ is only closed under topological minors.*

The result follows from Theorem 9.1 and the following structural result about graph classes with large tree-width established by Robertson and Seymour [**44**].

THEOREM 9.3 (Excluded Grid Theorem [**44**]). *There is a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that for all $k \geq 0$, every graph of tree-width at least $f(k)$ contains a $(k \times k)$-grid as a minor.*

It follows that if $\mathcal{C}$ is closed under minors and has unbounded tree-width, then the Excluded Grid Theorem implies that $\mathcal{G}^* \subseteq \mathcal{C}$. Intractability of MSO_2 on $\mathcal{C}$ therefore follows from Theorem 9.1. The generalisation to topological minors can be proved along the lines using *walls* instead of grids.

However, another consequence of the excluded grid theorem is that any (topological) minor closed class $\mathcal{C}$ of graphs of unbounded tree-width has very large tree-width, as it contains all grids and therefore graphs whose tree-width is roughly the square root of their order. Hence, there is a very large gap between the classes of graphs of bounded tree-width to which Courcelle's tractability results apply and the lower bound provided by Theorem 9.2.

To close this gap we will establish lower bounds for classes $\mathcal{C}$ of graphs of unbounded tree-width. Towards this aim we first need to measure the degree of unboundedness of the tree-width of classes $\mathcal{C}$ of graphs. We will do so by relating the tree-width of a graph in $\mathcal{C}$ to its order.

Definition 9.4. *Let σ be a binary signature. Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $p(n)$ be a polynomial.*

The tree-width of a class $\mathcal{C}$ of σ-structures is (f, p)-unbounded, *if for all $n \geq 0$*

(1) there is a graph $G_n \in \mathcal{C}$ of tree-width $\text{tw}(G_n)$ between n and $p(n)$ such that $\text{tw}(G_n) > f(|G|)$ and
(2) given n, G_n can be constructed in time $2^{n^{\varepsilon}}$, for some $\varepsilon < 1$.

The degree of $p(n)$ is called the gap degree. *The tree-width of $\mathcal{C}$ is* poly-logarithmically unbounded *if there are polynomials $p_i(n)$, $i \geq 0$, so that $\mathcal{C}$ is $(\log^i, p_i)$-unbounded for all i.*

The next theorem shows that essentially MSO_2 model-checking is fixed-parameter intractable on any class of graphs closed under sub-graphs with logarithmic tree-width. A similar result for classes of coloured graphs (but not closed under sub-graphs) was obtained in [**30**].

Theorem 9.5. [**35, 34**] *Let $\mathcal{C}$ be a class of graphs closed under sub-graphs, i.e. $G \in \mathcal{C}$ and $H \subseteq G$ implies $H \in \mathcal{C}$.*

(1) If the tree-width of $\mathcal{C}$ is $(\log^{28\gamma} n, p(n))$-unbounded, where p is a polynomial and $\gamma > 1$ is larger than the gap-degree of $\mathcal{C}$, then $\text{MC}(\text{MSO}_2, \mathcal{C})$ is not fpt unless Sat *can be solved in sub-exponential time $2^{o(n)}$.*
(2) If the tree-width of $\mathcal{C}$ is poly-logarithmically unbounded then $\text{MC}(\text{MSO}_2, \mathcal{C})$ is not fpt unless all problems in the polynomial-time hierarchy can be solved in sub-exponential time.

At its very core, the proof of the previous result also relies on a definition of large grids in graphs $G \in \mathcal{C}$. However, as the tree-with of graphs in $\mathcal{C}$ is only logarithmic in their order, the excluded grid theorem only yields grids of double logarithmic size which is not good enough. Instead the proof uses a new replacement structure for grids, called *grid-like minors* developed by Reed and Wood [**43**]. These structures do not exist in the graphs $G \in \mathcal{C}$ itself but only in certain intersection graphs of paths in G which makes their definition in MSO much more complicated. See [**34**] for details.

The previous results narrow the gap to Courcelle's theorem significantly. But clearly there still is a gap, between classes of bounded tree-width and those of super-logarithmic tree-width. In [**37**], Makowsky and Mariño exhibit a class of graphs of logarithmic tree-width which is closed under sub-graphs and on which MSO_2 model-checking becomes tractable. So there is no hope to improve the results in the previous theorem to classes with sub-logarithmic tree-width.

All previous results refer to classes which are closed under sub-graphs (or allow colourings which in some sense amounts to the same thing). We have seen that MSO_1 is fixed-parameter tractable even on classes of bounded clique-width. As clique-width is not closed under sub-graphs, one might wonder if even MSO_2 could be tractable on such classes. The question was answered in the negative by Courcelle et al. in [**4**] who showed that MSO_2 model-checking is not even tractable on the class of cliques, unless Exptime = NExptime. The model checking problem on the class of cliques might be considered as being slightly artificial. It is worth noticing, therefore, that the observation that MSO_2 is not tractable on classes of bounded clique-width has subsequently been observed also in purely algorithmic form [**22**] on graph classes of bounded clique-width. In particular, they show that problems such as HAMILTONIAN PATH, which are MSO_2 but not MSO_1 definable, are W[1]-hard when parameterized by the clique-width.

10. Further results on lower bounds

We close this part by commenting on lower bounds for first-order logic. It was shown in [**31**] that if a class $\mathcal{C}$ of graphs is closed under sub-graphs and not nowhere dense, then it has intractable first-order model-checking (subject to some technical condition). A class of graphs is *nowhere dense* if for every $r \geq 0$ there is a graph H_r such that $H_r \not\preccurlyeq_r G$ for all $G \in \mathcal{C}$. Nowhere dense classes of graphs are slightly more general than classes of bounded expansion considered in Section 8. Hence, there is again a gap between the lower and upper bound for first-order logic.

Finally, very little is known about lower bounds for MSO_1. Again, if $\mathcal{C}$ has unbounded tree-width and is closed under minors or topological minors then it has intractable model-checking (unless $P = \mathrm{NP}$). To obtain similar results as Theorem 9.5, we would first have to find an analogue of grid-like minors but to date not even a good candidate is known. Hence, we first need to understand obstructions for rank- and clique-width much better before any lower bounds can be shown.

References

[1] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small tree-width. *SIAM Journal on Computing*, 25:1305 – 1317, 1996.

[2] Julius R. Büchi. Weak second-order arithmetic and finite automata. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.

[3] Bruno Courcelle. Graph rewriting: An algebraic and logic approach. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume 2, pages 194 – 242. Elsevier, 1990.

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

[5] Bruno Courcelle and Johann A. Makowsky. Fusion in relational structures and the verification of monadic second-order properties. *Mathematical Structures in Computer Science*, 12(2):203–235, 2002.

[6] Anuj Dawar, Martin Grohe, and Stephan Kreutzer. Locally excluding a minor. In *Logic in Computer Science (LICS)*, pages 270–279, 2007.

[7] Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *46th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 637–646, 2005.

[8] Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken ichi Kawarabayashi. Decomposition, approximation, and coloring of odd-minor-free graphs. In *SODA*, pages 329–344, 2010.

[9] Matt DeVos, Guoli Ding, Bogdan Oporowski, DP. Sanders, Bruce Reed, Paul Seymour, and Dirk Vertigan. Excluding any graph as a minor allows a low tree-width 2-coloring. *Journal of Combinatorial Theory, Series B*, 91:25 – 41, 2004.

[10] Reinhard Diestel. *Graph Theory.* Springer-Verlag, 3rd edition, 2005.

[11] John Doner. Tree acceptors and some of their applications. *Journal of Computer and System Sciences*, 4:406–451, 1970.

[12] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity.* Springer, 1998.

[13] Zdeněk Dvořák, Daniel Král, and Robin Thomas. Deciding first-order properties for sparse graphs. In *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2010.

[14] Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. *Mathematical Logic.* Springer, 2nd edition, 1994.

[15] Michael Elberfeld, Andreas Jakoby, and Till Tantau. Logspace versions of the theorems of Bodlaender and Courcelle. In *Proceedings of the 51th Annual IEEE Symposium on Foundations of Computer Science*, pages 143–152, 2010.

[16] Calvin C. Elgot. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–51, 1961.

[17] David Eppstein. Subgraph isomorphism in planar graphs and related problems. *J. of Graph Algorithms and Applications*, 3(3):1–27, 1999.

[18] Solomon Feferman and Robert L. Vaught. The first order properties of products of algebraic systems. *Fundamenta Mathematicae*, 47:57–103, 1959.

[19] Jörg Flum, Markus Frick, and Martin Grohe. Query evaluation via tree-decompositions. *J. ACM*, 49(6):716–752, 2002.

[20] Jörg Flum and Martin Grohe. Fixed-parameter tractability, definability, and model checking. *SIAM Journal on Computing*, 31:113 – 145, 2001.

[21] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory.* Springer, 2006. ISBN 3-54-029952-1.

[22] Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Intractability of clique-width parameterizations. *SIAM Journal of Computing*, 39(5):1941–1956, 2010.

[23] Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable structures. *Journal of the ACM*, 48:1148 – 1206, 2001.

[24] 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.

[25] Haim Gaifman. On local and non-local properties. In J. Stern, editor, *Herbrand Symposium, Logic Colloquium '81*, pages 105 – 135. North Holland, 1982.

[26] Martin Grohe. Logic, graphs, and algorithms. In E.Grädel T.Wilke J.Flum, editor, *Logic and Automata – History and Perspectives.* Amsterdam University Press, 2007.

[27] Wilfrid Hodges. *A shorter model theory.* Cambridge University Press, 1997.

[28] Alexandr V. Kostochka. The minimum Hadwiger number for graphs with a given mean degree of vertices. *Metody Diskret. Analiz.*, 38:37–58, 1982. in Russian.

[29] Alexandr V. Kostochka. Lower bound of the Hadwiger number of graphs by their average degree. *Combinatorica*, 4(4):307–316, 1984.

[30] Stephan Kreutzer. On the parameterised intractability of monadic second-order logic. In *Proc. of Computer Science Logic (CSL)*, 2009.

[31] Stephan Kreutzer. Algorithmic meta-theorems. In Javier Esparza, Christian Michaux, and Charles Steinhorn, editors, *Finite and Algorithmic Model Theory*, London Mathematical Society Lecture Note Series, chapter 5, pages 177–270. Cambridge University Press, 2011. a preliminary version is available at Electronic Colloquium on Computational Complexity (ECCC), TR09-147, http://www.eccc.uni-trier.de/report/2009/147.

[32] Stephan Kreutzer. On the parameterized intractability of monadic second-order logic. *CSL 2009 Special Issue in Logical Methods in Computer Science (LMCS)*, 2011.

[33] Stephan Kreutzer and Anuj Dawar. Parameterized complexity of first-order logic. *Electronic Colloquium on Computational Complexity (ECCC)*, 16:131, 2009.

[34] Stephan Kreutzer and Siamak Tazari. Lower bounds for the complexity of monadic second-order logic. In *Logic in Computer Science (LICS)*, 2010.

[35] Stephan Kreutzer and Siamak Tazari. On brambles, grid-like minors, and parameterized intractability of monadic second-order logic. In *Symposium on Discrete Algorithms (SODA)*, 2010.

[36] Johann A. Makowsky. Algorithmic aspects of the Feferman-Vaught theorem. *Annals of Pure and Applied Logic*, 126(1–3):159 – 213, 2004.

[37] Johann A. Makowsky and Julian Mariño. Tree-width and the monadic quantifier hierarchy. *Theor. Comput. Sci.*, 1(303):157–170, 2003.

[38] Jaroslav Nesetril and Patrice Ossona de Mendez. Grad and classes with bounded expansion i. decompositions. *Eur. J. Comb.*, 29(3):760–776, 2008.

[39] Jaroslav Nesetril and Patrice Ossona de Mendez. Grad and classes with bounded expansion ii. algorithmic aspects. *Eur. J. Comb.*, 29(3):777–791, 2008.

[40] Jaroslav Nešetřil and Patrice Ossona de Mendez. Tree depth, subgraph coloring and homomorphisms. *European Journal of Combinatorics*, 2005.

[41] Jaroslav Nešetřil and Patrice Ossona de Mendez. On nowhere dense graphs. *European Journal of Combinatorics*, 2008. submitted.

[42] Jaroslav Nešetřil and Patrice Ossona de Mendez. First order properties on nowhere dense structures. *Journal of Symbolic Logic*, 75(3):868–887, 2010.

[43] Bruce Reed and Damian Wood. Polynomial treewidth forces a large grid-like minor. unpublished. Available at arXiv:0809.0724v3 [math.CO], 2008.

[44] Neil Robertson and Paul D. Seymour. Graph minors V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[45] Detlef Seese. Linear time computable problems and first-order descriptions. *Mathematical Structures in Computer Science*, 5:505–526, 1996.

[46] J. W. Thatcher and J. B. Wright. Generalised finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2:57–81, 1968.

[47] Andrew Thomason. An extremal function for contractions of graphs. *Math. Proc. Cambridge Philos. Soc.*, 95(2):261–265, 1984.

[48] Boris Trakhtenbrot. Finite automata and the logic of monadic predicates. *Doklady Akademii Nauk SSSR*, 140:326–329, 1961.

[49] Moshe Vardi. On the complexity of relational query languages. In *Proc. of the 14th Symposium on Theory of Computing (STOC)*, pages 137–146, 1982.

Lehrstuhl für Logik in der Informatik, Humboldt-Universität Berlin

Oxford University Computing Laboratory, Oxford, U.K. and Chair for Logic and Semantics, Technical University Berlin, Germany