Parameterized circuit complexity of model-checking on sparse structures[*]

Michał Pilipczuk, Sebastian Siebertz, and Szymon Toruńczyk
Institute of Informatics, University of Warsaw, Poland
{michal.pilipczuk,siebertz,szymtor}@mimuw.edu.pl

Abstract

We prove that for every class $\mathscr{C}$ of graphs with effectively bounded expansion, given a first-order sentence φ and an n-element structure $\mathbb{A}$ whose Gaifman graph belongs to $\mathscr{C}$, the question whether φ holds in $\mathbb{A}$ can be decided by a family of AC-circuits of size $f(\varphi) \cdot n^c$ and depth $f(\varphi) + c \log n$, where f is a computable function and c is a universal constant. This places the model-checking problem for classes of bounded expansion in the parameterized circuit complexity class para-AC^1. On the route to our result we prove that the basic decomposition toolbox for classes of bounded expansion, including orderings with bounded weak coloring numbers and low treedepth decompositions, can be computed in para-AC^1.

1 Introduction

Model-checking on sparse structures. We study the model-checking problem for first-order logic (FO): given a relational structure $\mathbb{A}$ and a first-order sentence φ over the vocabulary of $\mathbb{A}$, decide whether φ holds in $\mathbb{A}$. A naive algorithm for this problem recursively browses through all evaluations of each quantified variable and runs in time $n^{O(|\varphi|)}$, where n is the size of the universe of $\mathbb{A}$. Thus, the running time is polynomial for every fixed φ, but the degree of the polynomial depends on φ. In the language of parameterized complexity, this means that the model-checking problem for first-order logic on arbitrary structures is in the complexity class XP when parameterized by the input formula φ. This is traditionally put in contrast to the class FPT (for *fixed-parameter tractable*) where we require the existence of an algorithm with running time $f(\varphi) \cdot n^c$ for a computable function f and universal constant c; thus, the degree of the polynomial factor has to be independent of the parameter. See [9, 11, 16] for an introduction to parameterized complexity.

In general structures we do not hope for an FPT algorithm for model-checking FO, because the problem is complete for the class AW[$\star$] (cf. [16]). Already the problem of deciding the existence of a clique of size k in a given graph of size n, which is easily expressible by a first-order formula with k existential quantifiers, is W[1] hard in general, so believed not to be FPT when parametrized by k, i.e., solvable by an algorithm with running time $f(k) \cdot n^c$ for some computable f and fixed c. However, it was realized that on *sparse* structures, i.e. those whose Gaifman graph is sparse, efficient parameterized algorithms for model-checking FO exist. Starting with the result of Seese [30], who gave an FPT algorithm on structures with universally bounded degree, a long line of research focused on showing fixed-parameterized tractability of the problem for more and more general classes of sparse structures: of bounded local treewidth [17] (this includes planar and bounded-genus structures), excluding a fixed minor [15], and locally excluding a minor [10].

This line of research naturally converged to studying abstract notions of sparsity: classes of *bounded expansion* and *nowhere dense* classes. These two concepts form foundations of a deep and rapidly developing theory of sparse graph classes, first introduced and pursued by Nešetřil and Ossona de Mendez [23–26], which by now has found multiple applications in combinatorics, algorithm design, and logic. We refer the reader to the book of Nešetřil and Ossona de Mendez [27] for a comprehensive overview of the field as of 2012, and to the lecture notes of the first two authors for a compact and updated exposition of the basic toolbox [29].

Formally, a graph H is a *depth-r minor* of a graph G if H can be obtained from a subgraph of G by contracting mutually disjoint connected subgraphs of radius at most r. A class of graphs $\mathscr{C}$ has *bounded expansion* if there is a function $f\colon \mathbb{N} \to \mathbb{N}$ such that for every $r \in \mathbb{N}$, in every depth-r minor of a graph from $\mathscr{C}$ the ratio between the number of edges and the number of vertices is bounded by $f(r)$. More generally, $\mathscr{C}$ is *nowhere dense* if there is a function $t\colon \mathbb{N} \to \mathbb{N}$ such that no graph from $\mathscr{C}$ admits the clique $K_{t(r)}$ as a depth-r minor. Class $\mathscr{C}$ has *effectively bounded expansion*, respectively is *effectively nowhere dense*, if the respective function f or t as above is computable. These definitions are naturally generalized to classes of relational structures by considering the Gaifman graph of a structure.

Many classes of sparse graphs studied in the literature have (effectively) bounded expansion. These include: planar graphs, graphs of bounded maximum degree, graphs of bounded treewidth, and more generally, graphs excluding a fixed (topological) minor. A notable negative example is that classes with bounded *degeneracy*, equivalently with bounded *arboricity*, do not necessarily have bounded expansion, as there we have only a finite bound on the edge density in subgraphs (aka depth-0 minors). Every class of bounded expansion is nowhere dense, but the converse does not necessarily hold [25].

By the result of Dvořák, Král', and Thomas [13], the FO model-checking problem on any class $\mathscr{C}$ of effectively bounded expansion admits a linear FPT algorithm, i.e. with running time $f(\varphi) \cdot n$ for computable f. Grohe, Kreutzer, and the second author [21] lifted this result to any effectively nowhere dense class $\mathscr{C}$; here, the dependence on the structure size n is *almost linear*, i.e. of the form

[*]The work of M. Pilipczuk and S. Siebertz is supported by the National Science Centre of Poland via POLONEZ grant agreement UMO-2015/19/P/ST6/03998, which has received funding from the European Union's Horizon 2020 research and innovation programme (Marie Skłodowska-Curie grant agreement No. 665778). The work of Sz. Toruńczyk is supported by the National Science Centre of Poland grant 2016/21/D/ST6/01485.

LICS '18, July 9–12, 2018, Oxford, United Kingdom
© 2018 Copyright held by the owner/author(s). Publication rights licensed to the Association for Computing Machinery.
ACM ISBN 978-1-4503-5583-4/18/07…$0.00
https://doi.org/10.1145/3209108.3209136

$n^{1+\varepsilon}$ for any $\varepsilon > 0$. As observed by Dvořák et al. [13], the result of Grohe et al. [21] is the final answer as long as subgraph-closed classes are concerned: on any subgraph-closed class $\mathscr{C}$ that is not nowhere dense, the FO model-checking problem is already AW[$\star$]-complete. Conceptually, this means that the notion of nowhere denseness exactly characterizes classes of inputs where sparsity-based arguments can lead to efficient parameterized algorithms for deciding first-order definable properties.

Parameterized circuit complexity. In this paper we take a different angle on the parameterized complexity of model-checking FO, namely that of *circuit complexity.* A fundamental fact from descriptive complexity is that FO is essentially equivalent to AC^0 (c.f. [22] for a precise statement). In particular, every fixed first-order expressible property can be checked by a family of AC-circuits of polynomial size and constant depth. More precisely, provided the property is expressed by a first-order sentence φ, the circuit for n-element inputs has size $n^{O(|\varphi|)}$ and depth $O(|\varphi|)$. Viewing this via the standard interpretation of circuits as an abstraction for parallel algorithms, this is a highly parallelizable algorithm performing total (sequential) XP work. Obviously, in general we cannot expect the problem to be solvable by circuits of FPT size (i.e. of size $f(\varphi) \cdot n^c$ for computable f and constant c), as evaluating such circuits would yield a sequential FPT algorithm, implying FPT = AW[$\star$]. However, the question for known classes for which FO model-checking is FPT persists: how, and in what sense, can we solve FO model-checking on these classes using circuits of FPT size and low depth? Viewing circuits again as a model for parallelization, this would correspond to a well-parallelizable FPT algorithm.

Curiously, even though the complexity-theoretical foundations of parameterized complexity are expressed using circuit complexity, the question of what are the appropriate analogues of standard circuit complexity classes in parameterized complexity was not systematically studied up to very recently, when Elberfeld et al. [14] and Bannach et al. [3] introduced an appropriate definitional layer and gave several foundational results. Slightly informally, a parameterized problem is in the class para-AC^i (where $i > 0$) if it can be solved by an (appropriately uniform) family of AC-circuits $(C_{n,k})_{n,k\in\mathbb{N}}$, where the circuit $C_{n,k}$ solves the problem on inputs of size n and parameter value k, such that each $C_{n,k}$ has size $f(k) \cdot n^c$ and depth $f(k) + c \cdot \log^i n$, for a computable function f and universal constant c. The classes para-NC^i are defined similarly using NC-circuits. The class para-AC^0 is defined slightly differently: we require the depth to be bounded by a universal constant, independent of the parameter. By allowing the depth to be bounded by a function of the parameter we obtain the larger class para-$\mathrm{AC}^{0\uparrow}$. We give the formal definitions and fix the notation in Section 2.

In [3] Bannach et al. showed how the technique of *color coding* can be implemented in para-AC^0, leading to a first batch of results for several parameterized problems. Later, Bannach and Tantau [4] showed that model-checking monadic second-order logic (MSO) can be done in para-$\mathrm{AC}^{0\uparrow}$ on structures of bounded treedepth and in para-$\mathrm{NC}^{2+\varepsilon}$ for any $\varepsilon > 0$ on structures of bounded treewidth; here, the parameter is both the formula and the treedepth, respectively the treewidth of the input structure. This is a parameterized circuit complexity analogue of the classic theorem of Courcelle stating that model-checking MSO is fixed-parameter tractable when parameterized by the formula and the treewidth of the input structure. Recent advances show descriptive relations between parameterized circuit complexity and fragments of FO with a bounded number of variables [8], and applications to kernelization [5].

In this light, it is natural to ask about the parameterized circuit complexity of model checking FO on sparse structures. Investigating this question is precisely the goal of this work. Our main result is encompassed by the following theorem.

Theorem 1.1. *Suppose $\mathscr{C}$ is a graph class with effectively bounded expansion and let Σ be a relational vocabulary of arity 2. Then the following problem parameterized by $\varphi \in \mathrm{FO}[\Sigma]$ is in* para-AC^1*: given a Σ-structure $\mathbb{A}$ whose Gaifman graph belongs to $\mathscr{C}$, determine whether $\mathbb{A} \models \varphi$.*

Unraveling the definitions, Theorem 1.1 states that model-checking FO on Σ-structures with Gaifman graphs belonging to $\mathscr{C}$ can be done using a family of AC circuits $(C_{n,\varphi})_{n\in\mathbb{N},\varphi\in\mathrm{FO}[\Sigma]}$, where $C_{n,\varphi}$ verifies the satisfaction of φ on structures with n elements, and each $C_{n,\varphi}$ has size $f(\varphi) \cdot n^c$ and depth $f(\varphi) + c \log n$, for a computable function f and universal constant c. Viewing circuits as an abstraction for parallel algorithms, this means that the problem can be solved in parallel time $f(\varphi) + c \log n$ and performing total work $f(\varphi) \cdot n^c$. Hence Theorem 1.1 can be regarded as a parallelized variant of the result of Dvořák et al. [13].

The assumption in Theorem 1.1 that Σ has arity 2 allows us to abstract away the question of how the input is represented, as we simply assume that each relation in $\mathbb{A}$ is encoded on input as a one- or two-dimensional boolean table. In the presence of higher-arity relations, the choice of an encoding could influence the statements about bounds on circuit sizes in a technical way. We prefer to avoid these issues and simply assume that there are no higher-arity relations.

Our techniques. We prove Theorem 1.1 by analyzing the existing approach to proving fixed-parameter tractability in the sequential case. Essentially, the idea is to first compute a suitable decomposition of the input structure, and then leverage this decomposition to give a quantifier elimination procedure for first-order logic. A suitable decomposition has the form of a *low treedepth coloring* that uses a bounded number of colors; it is known that such colorings exist for graphs from classes of bounded expansion [23]. Efficient algorithms for computing low treedepth colorings provided by Nešetřil and Ossona de Mendez [23] allowed Dvořák et al. [13] to give an efficient quantifier elimination procedure that reduces every first-order formula to a quantifier-free formula at the cost of extending the structure by adding new unary relations and unary functions, which however do not change the Gaifman graph. From this, an algorithm for model-checking follows. Later, Grohe and Kreutzer [19] gave a new presentation of the quantifier elimination procedure, which is conceptually quite different from the original argument of [13]. In particular it reduces every formula to an existential formula instead of a quantifier-free one, but the extension of the structure does not use function symbols.

Our work toward the proof of Theorem 1.1 is divided into two parts. First, we prove that a low treedepth coloring of the graph can be computed in para-AC^1. Second, using this result we revisit the quantifier elimination procedure and show that it can be implemented in para-AC^1.

For computing a low treedepth coloring, on high level we follow the standard approach: first find a vertex ordering of the given graph with bounded *weak coloring number*, and then apply a coloring procedure on this vertex ordering to get a low treedepth coloring. Classic implementations of both these steps are sequential, however we show that both of them can be performed in para-AC^1. For the first step, the main idea is to construct the ordering by extracting the vertices not one by one, as a sequential algorithm would do, but in much larger chunks. Namely, we perform $\log n$ rounds where in each round at least half of the remaining vertices are extracted and ordered, which directly translates to a construction of a para-AC^1 circuit family. This comes at a price in the quality of the obtained ordering; in other words, we trade the approximation factor of the algorithm for its parallelization. For the second step, we follow the classic divide-and-conquer approach to parallel coloring graphs of bounded maximum degree. Then we extend this to graphs of bounded degeneracy by applying essentially the same technique of dividing the graph into $\log n$ parts, each inducing a graph of bounded maximum degree.

We remark that the approach explained above is heavily inspired by the existing body of work on *distributed algorithms* for sparse graphs. We relied on ideas from Barenboim and Elkin [7] who, among other results, gave an $O(\log n)$-time distributed algorithm that, given a graph of degeneracy d, finds its proper coloring with $O(d^2)$ colors[1]. In particular, Barenboim and Elkin showed how to compute an approximate degeneracy ordering of such a graph in $O(\log n)$ communication rounds by extracting half of the remaining vertices in each round; our algorithm for this step is a circuit implementation of this procedure, lifted to the weak coloring number instead of degeneracy. Using the results of [7] and the approach via fraternal augmentations, Nešetřil and Ossona de Mendez [28] gave a logarithmic-time distributed algorithm that, given a graph from a fixed class of bounded expansion, computes its treedepth-p coloring using a constant number of colors, for any constant p.

For the quantifier elimination procedure, we essentially revisit the existing approach and show that it can be implemented in para-AC^1. This requires technical attention in several places, but conceptually there is no new ingredient. Our argument is roughly based on the exposition of Grohe and Kreutzer [19]. However, we obtain a stronger final form, similar to that of Dvořák et al. [13], replace the usage of FO types with an explicit combinatorial argument in the spirit of marking witnesses as in [13], and streamline the presentation.

Due to space constraints some proofs (marked with $\star$) are not presented in this extended abstract and can be found in the full version of the paper.

2 Preliminaries

All graphs considered in this paper are simple, i.e. do not contain self-loops or multiple edges connecting the same pair of vertices. We use standard graph notation, see e.g. [9].

We explain the definitional layer of parameterized circuit complexity basing our notation on Bannach et al. [3], though prior foundational work on parameterized circuit complexity was done by Elberfeld et al. [14]. An AC-*circuit* C is a directed acyclic graph with node set consisting of input, conjunction (AND), disjunction (OR), and negation (NOT) gates. There are no restrictions on the fan-in or fan-out of the gates. One or more sources of C are designated as the output gates, and both input and output gates of C are ordered. If $(u_1, \ldots, u_n)$ and $(v_1, \ldots, v_m)$ are the input and output gates of C, respectively, then C evaluates a function from $\{0,1\}^n$ to $\{0,1\}^m$ defined as follows: given input $x = (x_1, \ldots, x_n)$, set the value of each input gate u_i to x_i, evaluate the gates of the circuit in a bottom-up manner naturally, and define the output y to be the sequence of values computed in gates $(v_1, \ldots, v_m)$. The *depth* of a circuit is the length of a longest path from an input gate to an output gate. The *size* of a circuit is the number of its gates.

A *parameterized transformation* is a function $F\colon \{0,1\}^\star \to \{0,1\}^\star$ together with a polynomial-time computable function $\kappa\colon \{0,1\}^\star \to \mathbb{I}$, called *parameterization.* Here $\mathbb{I}$ is some indexing set for parameters and we assume that its elements can be encoded as binary strings. A *parameterized problem* is just a parameterized transformation with the output always belonging to $\{0,1\}$, for false and true, respectively.

A parameterized transformation is in the class FPT if there is an algorithm that computes it in time $f(k) \cdot n^c$ on inputs of size n and parameter value k, where f is a computable function and c is a universal constant. It is in the class linFPT if moreover $c = 1$ and $\kappa(\cdot)$ is linear-time computable.

For an indexing set $\mathbb{I}$, we may consider a family $(C_{n,k})_{n\in\mathbb{N}, k\in\mathbb{I}}$ of AC-circuits, where each $C_{n,k}$ has exactly n inputs. We say that such a family is $\mathsf{dlogtime}$-*uniform* if there exists an algorithm that given $n \in \mathbb{N}$, $k \in \mathbb{I}$, and $i \in \mathbb{N}$, all encoded in binary, computes the i-th bit of the encoding of $C_{n,k}$ in time $f(k) + O(\log i + \log n)$, for some computable function f. All circuit families in this paper are $\mathsf{dlogtime}$-uniform; this will always follow from the construction in a straightforward manner, so we refrain from providing technical details in order not to obfuscate the main ideas.

For $i > 0$, we say that a parameterized transformation (F, κ) is in para-AC^i if there exists a $\mathsf{dlogtime}$-uniform family of circuits $(C_{n,k})_{n\in\mathbb{N}, k\in\mathbb{I}}$ such that (a) $C_{n,k}$ has size $f(k) \cdot n^{O(1)}$ and depth $f(k) + O(\log^i n)$ for some computable function $f\colon \mathbb{I} \to \mathbb{N}$; (b) for each $x \in \{0,1\}^\star$, the output of $C_{|x|,\kappa(x)}$ applied to x is $F(x)$. Note that the above definition implicitly assumes that for all x, x' with $|x| = |x'|$ and $\kappa(x) = \kappa(x')$, the outputs $F(x)$ and $F(x')$ have the same length, as this length must be equal to the number of outputs of $C_{|x|,\kappa(x)}$. Bannach et al. [3] also define a larger class para-$\mathsf{AC}^{i\uparrow}$ by relaxing the restriction on the depth from $f(k) + O(\log^i n)$ to $f(k) \cdot \log^i n$, for a computable function f. It is easy to see that para-$\mathsf{AC}^i \subseteq$ para-$\mathsf{AC}^{i\uparrow} \subseteq$ para-$\mathsf{AC}^{i+\varepsilon}$ for any $\varepsilon > 0$.

For $i = 0$, the classes para-AC^0 and para-$\mathsf{AC}^{0\uparrow}$ are defined slightly differently: in para-AC^0 we require that the depth of the circuits is $O(1)$, i.e. bounded by a universal constant independent of the parameter, while in para-$\mathsf{AC}^{0\uparrow}$ we allow the depth of $C_{n,k}$ to be bounded by $f(k)$, for a computable function f.

Let us briefly elaborate on the differences between the classes para-AC^i and para-$\mathsf{AC}^{i\uparrow}$. Both definitions are natural candidates for what a parameterized analogue of AC^i should be, as in both cases the class becomes AC^i whenever k is fixed to be a constant. However, bounding the depth by $f(k) + c \cdot \log n^i$ instead of $f(k) \cdot \log n^i$ gives better guarantees when transforming the circuit to a formula (a circuit with maximum fan-out 1). For instance, every para-NC^1 circuit (where we restrict fan-in to be at most 2) can be unravelled to an equivalent para-NC^1 formula, which is the analogue of a well-known property of NC^1, but this is no longer the case for para-$\mathsf{NC}^{1\uparrow}$, because the formula size would be $n^{f(k)}$ instead of $f(k) \cdot n^{O(1)}$. Similarly, every para-AC^0 circuit can be unravelled to

[1] Barenboim and Elkin use the parameter *arboricity* which differs from degeneracy by multiplicative factor at most 2.

an equivalent para-AC^0 formula, but this property is not shared by para-AC$^{0\uparrow}$. Nevertheless, classes para-AC$^{i\uparrow}$ are still worth studying due to encompassing many natural algorithms. This state reflects the situation in parameterized analogues of nondeterministic logspace, where the difference between bounding the space by $f(k) + c \log n$ and by $f(k) \cdot \log n$ has dramatic implications for determinization results. We refer to the work of Elberfeld et al. [14] for a broader exposition of these connections.

Graph problems. Typically, throughout this paper the input to a parameterized transformation will be a graph G on n vertices. In this case we will always assume that the input is encoded as the $n \times n$ binary adjacency matrix of the graph; thus the circuit computing the transformation needs to have n^2 inputs. Abusing the above notation somewhat, for parameterized circuit classes, we will interpret the $|x|$ for an encoding x of G as the number n of vertices of G, instead of the actual length n^2 of x. Thus, the domain of a parameterized transformation defined on graphs consists of all words of length n^2 for some integer n, interpreted as adjacency matrices, and each circuit $C_{n,k}$ will actually have n^2 inputs. In case the input to the transformation consists of a graph together with some additional piece of information (e.g. additional numerical parameters, a coloring of the graph, or an ordering of its vertices), the appropriate encoding of this additional information (the form of which will be specified later) is provided via extra input gates. In all cases, the circuit $C_{n,k}$ will be responsible for the treatment of instances with n vertices and parameter value k.

Basic circuit constructions. One of the fundamental results for parameterized circuit complexity is that counting up to a threshold parameter d can be done in para-AC^0. More precisely, consider the problem Threshold parameterized by d: given a word $x \in \{0, 1\}^\star$, determine whether x has at most d ones. A naive construction of a constant-depth circuit would be to check every d-tuple of inputs, but this would result in a circuit of size $\Omega(n^d)$. However, Bannach et al. [3] showed that Threshold in fact is in para-AC^0 using a parallel implementation of color coding.

Theorem 2.1 (Lemma 3.3 of [3]). *Threshold is in* para-AC^0.

As a corollary, given a graph, we can G compute in para-AC^0 the set of vertices of G that have degree at most d.

Corollary 2.2 ($\star$). *The following transformation parameterized by d is in* para-AC^0*: given a graph G, compute the set of vertices of G that have degree at most d.*

Another primitive in our algorithms will be counting distances in graphs up to a fixed threshold $r \in \mathbb{N}$. This is encapsulated in the following lemma.

Lemma 2.3 ($\star$). *There exists a* dlogtime*-uniform family of* AC*-circuits $(D_{n,r})_{n,r\in\mathbb{N}}$ such that each $D_{n,r}$, given a n-vertex graph G, outputs the $n \times n$ boolean matrix encoding, for each pair of vertices u, v of G, whether the distance between u and v in G is at most r. Each $D_{n,r}$ has size $n^{O(1)}$ and depth $O(\log r)$.*

Proof sketch. Let A be the adjacency matrix of G with ones added on the diagonal. Then it suffices to compute A^r, the rth boolean power of A, that is, its rth power in the (OR, AND)-semiring over $\{0, 1\}$. Observe that A^r can be computed from A by a circuit of size $n^{O(1)}$ and depth $O(\log r)$ using the iterative squaring algorithm. □

3 Bounded degeneracy

In this section we study the case of graphs of degeneracy d. Those are graphs which can be linearly ordered in such a way that every vertex has at most d smaller neighbors. It is well known that a class C of graphs has degeneracy bounded by some constant d if and only if there is a number c such that in every subgraph H of a graph $G \in C$, the ratio between the number of edges and number of vertices in H is bounded by c. Therefore, bounded degeneracy is similar to bounded expansion, but we only bound the density of depth-0 minors, i.e., subgraphs. Many proof techniques concerning bounded expansion classes stem from the techniques for classes of bounded degeneracy. This is no different in our paper. In this section, we prove some parallelized variants of known results for classes of bounded degeneracy. Specifically, it is known that graphs of degeneracy d admit a proper coloring using $d + 1$ colors, and in Lemma 3.6, we show that a coloring using $O(d^2)$ colors can be computed in para-AC^1. In the following section, we extend this result to classes of bounded expansion.

Definition and basic properties. A *vertex ordering* of a graph G is any ordering $\sigma = (v_1, \ldots, v_n)$ of the vertices of G. A vertex ordering σ can be also understood via the linear order $\leqslant_\sigma$ on $V(G)$ imposed by it: $u \leqslant_\sigma v$ iff $u = v$ or u appears earlier than v in σ. Whenever a vertex ordering of an n-vertex graph is represented in a circuit construction, we assume that it is represented as the $n \times n$ boolean matrix encoding the order $\leqslant_\sigma$. The *degeneracy* of the ordering σ is the least d such that every vertex $v \in V(G)$ has at most d neighbors u satisfying $u <_\sigma v$. The *degeneracy* of a graph is the smallest possible degeneracy of its vertex orderings.

We use the following fact that a graph of degeneracy d has a linear number of edges, and moreover there are few vertices with degrees significantly larger than d.

Proposition 3.1 ($\star$). *An n-vertex graph G of degeneracy at most d has at most dn edges. Moreover, for every real $c \geqslant 1$, G has less than $\frac{n}{c}$ vertices of degree larger than $2cd$.*

Finally, we recall the well-known fact that a graph of degeneracy d admits a proper coloring with $d + 1$ colors. Recall here that a proper coloring of a graph is a coloring of its vertices such that no edge has both endpoints of the same color. Equivalently, every color class is an independent set.

Proposition 3.2. *A graph of degeneracy d admits a proper coloring with $d + 1$ colors.*

Proof. Let G be the graph in question and let σ be a vertex ordering of G of degeneracy d. Consider the following greedy procedure that colors vertices of G with colors $\{1, \ldots, d + 1\}$: iterate through vertices of G in the order of σ and for each vertex u assign to it any color that is not present among the neighbors of u smaller in σ. Since there are at most d such neighbors, such a color will always exist. □

Our goal in this section is to prove a parallelized variant of Proposition 3.2. This will be achieved in Lemma 3.6 below.

3.1 Block vertex orderings and computational aspects

We will use a relaxed variant of degeneracy orderings where vertices come in ordered blocks and every vertex has few neighbors in its own and smaller blocks.

Definition 3.3. A *block vertex ordering* of a graph G is an ordered partition $\tau = (B_1, B_2, \ldots, B_\ell)$ of the vertex set of G; its *length* is the number of blocks ℓ. The *degeneracy* of τ is the least integer d such that for each $i \in \{1, \ldots, \ell\}$, every vertex $v \in B_i$ has at most d neighbors in $\bigcup_{j=1}^{i} B_j$.

A block vertex ordering τ as above naturally imposes a total quasi-order $\leqslant_\tau$ on the vertex set of G: $u \leqslant_\tau v$ iff $u \in B_i$ and $v \in B_j$ with $i \leqslant j$. In our circuits we will assume that a block vertex ordering of an n-vertex graph is represented by the $n \times n$ boolean matrix encoding $\leqslant_\tau$.

Obviously, if $\sigma = (v_1, \ldots, v_n)$ is a vertex ordering of G, then the degeneracy of σ is equal to the degeneracy of the block vertex ordering $(\{v_1\}, \ldots, \{v_n\})$. On the other hand, if $\tau = (B_1, \ldots, B_\ell)$ is a block vertex ordering of G of degeneracy d, then by ordering each block arbitrarily and concatenating these orderings we obtain a vertex ordering of G of degeneracy at most d.

It will be important however that provided G has bounded degeneracy, we may find a block vertex ordering of small degeneracy that is of logarithmic length. This idea is also the cornerstone of the work of Barenboim and Elkin [7], who gave logarithmic-time distributed algorithms to approximately color graphs of bounded degeneracy. Our block vertex orderings correspond to *H-partitions* in their nomenclature, and similarly to us they show that an H-partition of small degeneracy and logarithmic size can be efficiently computed by repeatedly taking vertices of small degree. Actually, Barenboim and Elkin attribute the idea to an earlier work of Arikati et al. [1] that used the PRAM model of parallel algorithms.

Lemma 3.4. *The following transformation parameterized by d is in* para-AC^1*: Given an n-vertex graph G of degeneracy at most d, compute a block vertex ordering of G of degeneracy at most $4d$ and length at most $\log n$.*

Proof sketch. We describe combinatorially how the block vertex ordering is constructed. Starting with $G_0 = G$, define the last block to consist of all vertices of G_0 that have degree at most $4d$ in G_0; by Proposition 3.1 with $c = 2$, this block constitutes more than half of the vertex set of G_0. Remove the block from G_0 yielding a graph G_1 and apply again the same procedure to G_1. That is, define the second-to-last block to consist of all vertices of G_1 that have degree at most $4d$ in G_1 and remove this block yielding G_2; since G_1 is a subgraph of G, again Proposition 3.1 ensures us that in this manner we remove more than half of the remaining vertex set. Thus the construction finishes with an empty graph after at most $\log n$ iterations and yielding at most $\log n$ blocks in total. It is straightforward to see that the degeneracy of the obtained block vertex ordering is at most $4d$. It is easy to see that we can implement the above procedure using an AC-circuit family with prescribed size and depth constraints. □

By ordering arbitrarily each block of the block vertex ordering given by Lemma 3.4 we obtain the following corollary: given a graph G of degeneracy at most d, computing a vertex ordering of G of degeneracy at most $4d$ can be done in para-AC^1. In other words, this is a 4-approximation algorithm for degeneracy in para-AC^1, where the target degeneracy is the parameter.

It is natural to ask whether the following problem of determining degeneracy exactly is also in para-AC^1: for a parameter d, determine whether the degeneracy of a given graph is at most d. Recall here that this problem can be solved in polynomial time. We give a negative answer to this side question by proving the following theorem.

Theorem 3.5 ($\star$). *The following problem is* P*-hard under logspace reductions: Given a graph G, determine whether the degeneracy of G is at most* 2.

Thus, if determining degeneracy exactly was in para-AC^1, or even in para-AC^i for any i, then NC = P. Moreover, the same can be concluded about approximating degeneracy up to any factor $\alpha < \frac{3}{2}$.

3.2 Coloring graphs of bounded degeneracy

Recall from Proposition 3.2 that graphs of bounded degeneracy can be colored using a bounded number of colors. In this section, we show the following, parallelized variant of this result.

Lemma 3.6 ($\star$). *The following transformation parameterized by d is in* para-AC^1*: Given a graph of degeneracy at most d, compute some proper coloring with $(4d + 1)^2$ colors.*

Recall that a similar statement in the context of distributed computing was obtained by Barenboim and Elkin [7]. The rest of Section 3 is devoted to outlining a proof of Lemma 3.6.

We shall first present how to greedily color bounded degree graphs in para-AC^1, and then leverage this understanding to color graphs of bounded degeneracy in para-AC^1. But before this, we present a key technical lemma that will be used multiple times. Essentially it says that provided we have already achieved some proper coloring with h colors, we may then use it to compute a better coloring using a circuit of depth linear in h. This trick was also used by Barenboim and Elkin [7].

Lemma 3.7. *There exists a* dlogtime*-uniform family of* AC*-circuits $(K_{n,d,h})_{n,d,h \in \mathbb{N}}$ such that each $K_{n,d,h}$, given a n-vertex graph G together with its block vertex ordering $\tau = (B_1, \ldots, B_\ell)$ with $\ell \leqslant h$ and of degeneracy d with each block B_i being an independent set in G, computes a proper coloring of G with $d + 1$ colors. Each $K_{n,d,h}$ has size $n^{O(1)}$ and depth $O(h)$.*

Proof sketch. We build a coloring $\lambda\colon V(G) \to \{1, \ldots, d + 1\}$ in h rounds, where in round i all vertices from block B_i receive their colors in λ. In round i every vertex $u \in B_i$ inspects all its neighbors and sets its own color to be the smallest color that is not present among its neighbors residing in lower blocks. Note that such a color always exists since the number of neighbors is at most d, by the assumption about the degeneracy of the input block vertex ordering, and there are $d + 1$ colors available. To see that λ constructed in this way will be a proper coloring of G, observe that every edge uv of G connects two vertices from different blocks, say $u \in B_i$ and $v \in B_j$ for $i < j$. Hence v will pick its color in λ to be different than $\lambda(u)$. It is easy to implement this procedure by an AC-circuit of polynomial size and depth $O(h)$. □

Corollary 3.8 ($\star$). *There is a* dlogtime*-uniform family of* AC*-circuits $(L_{n,d,h})_{n,d,h \in \mathbb{N}}$ such that each $L_{n,d,h}$, given a n-vertex graph G of maximum degree at most d together with its proper coloring with h colors, computes a proper coloring of G with $d + 1$ colors. Each $L_{n,d,h}$ has size $n^{O(1)}$ and depth $O(h)$.*

Our first step towards Lemma 3.6 is the treatment of graphs of bounded degree. A graph of maximum degree at most Δ can be greedily colored with $\Delta + 1$ colors. A naive implementation of this greedy procedure is sequential, but we will show now how to perform this task in para-AC^1. We remark that finding optimum or near-optimum proper colorings of graphs of bounded maximum degree is a very classic topic in distributed computing with a vast existing literature. We refer to the work of Barenboim [6] for the currently fastest algorithms and an excellent overview of the area.

We first show a weaker result, namely that a proper coloring with $\Delta + 1$ colors of a graph of maximum degree at most Δ can be computed in para-AC$^{1\uparrow}$, when parameterized by Δ. Recall that this means that we allow depth $f(\Delta) \cdot \log n$ instead of $f(\Delta) + O(\log n)$. This can be done using a simple Divide&Conquer trick, which dates back to a classic $O(\Delta \log n)$-time distributed algorithm for this problem of Goldberg et al. [18] (see also [2]).

Lemma 3.9. *The following transformation parameterized by Δ is in* para-AC$^{1\uparrow}$*: Given a graph of maximum degree at most Δ, compute some proper coloring with $\Delta + 1$ colors.*

Proof sketch. We perform a divide-and-conquer algorithm. Given the input graph G with n vertices, arbitrarily partition its vertex set into two subset V_1 and V_2, each of size at most $\lceil n/2 \rceil$. Let G_1 and G_2 be the subgraphs induced by V_1 and V_2 in G, respectively. Each of G_1, G_2 has maximum degree at most Δ, hence we can apply the algorithm recursively to both these graphs, yielding proper colorings λ_1, λ_2 of G_1 and G_2, respectively, each using $\Delta + 1$ colors. By taking the union of these two colorings, where colors from different subgraphs are considered different, we obtain a proper coloring of G with $2\Delta + 2$ colors. We may now apply the circuit given by Corollary 3.8 for $h = 2\Delta + 2$ to compute a coloring of G with $\Delta + 1$ colors. Again it is easy to turn the above algorithm into a circuit. □

We now show containment in para-AC^1.

Lemma 3.10. *The following transformation parameterized by Δ is in* para-AC^1*: Given a graph of maximum degree at most Δ, compute some proper coloring with $\Delta + 1$ colors.*

Proof. Let every vertex u of G arbitrarily (say, according to the order of inputs) put numbers $1, \ldots, \deg(u)$ on edges incident to it; thus every edge is labelled with two numbers, one originating from each endpoint. Such a labeling can be computed by a circuit of size $f(\Delta) \cdot n^{O(1)}$ and depth $O(1)$ using the circuits provided by Theorem 2.1 to count, for every neighbor v of u, the number of neighbors of u with smaller indices than v.

For every pair of indices (i, j) with $1 \leqslant i \leqslant j \leqslant \Delta$, let $G_{i,j}$ be the subgraph of G with $V(G_{i,j}) = V(G)$ and $E(G_{i,j})$ consisting of those edges e of G, for which one endpoint of e labelled e with i, and the second labelled it with j. Observe that the maximum degree of $G_{i,j}$ is at most 2, since every vertex u of G can be adjacent to at most two edges of $G_{i,j}$: the one it labelled with i and the one it labelled with j. Using Lemma 3.9 we can compute, for each $1 \leqslant i \leqslant j \leqslant \Delta$, a proper 3-coloring $\lambda_{i,j}$ of $G_{i,j}$ using an AC-circuit of size $n^{O(1)}$ and depth $O(\log n)$. Next we construct a product coloring λ of G with $3^{\binom{\Delta+1}{2}}$ colors: a color of a vertex u in λ is the $\binom{\Delta+1}{2}$-tuple of colors of u in the colorings $\lambda_{i,j}$ for all $1 \leqslant i \leqslant j \leqslant \Delta$. Since each edge of G participates in exactly one of subgraphs $G_{i,j}$, it is clear that λ is a proper coloring of G. We may finally apply Corollary 3.8 for $h = 3^{\binom{\Delta+1}{2}}$ to compute a proper coloring of G with $\Delta + 1$ colors using an AC-circuit of size $n^{O(1)}$ and depth $O(3^{\binom{\Delta+1}{2}})$. Thus, in total we have constructed a circuit of size $f(\Delta) \cdot n^{O(1)}$ and depth $O(3^{\binom{\Delta+1}{2}} + \log n)$. □

We finally have all the tools to prove Lemma 3.6.

Proof of Lemma 3.6. By Lemma 3.4 we may compute a block vertex ordering τ of G of degeneracy at most $4d$ and length at most $\log n$ in para-AC^1. Partition the edges of G into two graphs G_1, G_2 on the same vertex set as G: the edge set of G_1 consists of all edges whose endpoints lie in the same block of τ, while the edge set of G_2 consists of all edges whose endpoints lie in different blocks of τ. Observe that G_1 is a graph of maximum degree at most $4d$, hence we may apply Lemma 3.10 to compute some proper coloring λ_1 with $4d + 1$ colors in para-AC^1. On the other hand, τ is a block vertex ordering of G_2 with degeneracy at most $4d$, length at most $\log n$, and every block being an independent set in G_2. Hence, we may apply Lemma 3.7 to compute a proper coloring λ_2 of G_2 with $4d + 1$ colors using a circuit of polynomial size and depth $O(\log n)$. Finally, let λ be the product coloring of λ_1 and λ_2: the color a vertex u receives in λ is the pair of colors it received in λ_1 and λ_2. Since each edge of G participates either in G_1 or in G_2, λ constructed in this manner is a proper coloring of G with $(4d+1)^2$ colors. The fact that the constructed circuit satisfies the required size and depth bounds follows directly from the construction and from the bounds provided by Lemma 3.4, Lemma 3.7, and Lemma 3.10. □

4 Computing low treedepth colorings

The fact that graphs of bounded degeneracy admit a proper coloring using a bounded number of colors can be generalized to graphs of bounded expansion, using the notion of *low treedepth colorings*. Intuitively, for a fixed $p \in \mathbb{N}$, a *treedepth-p coloring* is a coloring using a bounded number of colors, such that any p color classes induce a graph which has a depth-first search (DFS) forest of bounded depth. Such colorings turn out to be very useful for many algorithmic purposes, notably, for model-checking. In this section, we generalize the result from the previous section, and show that such colorings can be computed in para-AC^1. As previously, such colorings are obtained by first finding an appropriate ordering of the graph, related to the notion of *weak reachability*, which we recall below.

4.1 Weak coloring number

Suppose G is a graph, $r \in \mathbb{N}$, and σ is a vertex ordering of G. For two vertices $u, v \in V(G)$ with $u \leqslant_\sigma v$, we say that u is *weakly r-reachable* from v if there is a path of length r in G that leads from v to u and whose internal vertices are all larger than u in σ. The set of vertices weakly r-reachable from v in σ is denoted by $\mathrm{WReach}_r[G, \sigma, v]$. The *weak r-coloring number* of σ is equal to

$$\mathrm{wcol}_r(G, \sigma) = \max_{v \in V(G)} |\mathrm{WReach}_r[G, \sigma, v]|$$

and the weak r-coloring number of G, denoted $\mathrm{wcol}(G)$, is the smallest weak r-coloring number of a vertex ordering of G. The following theorem is the main result of this section.

Theorem 4.1 (⋆). *The following transformation parameterized by integers r and d is in* para-AC^1*: Given a graph G whose all depth-$(r-1)$ have edge density at most d, compute a vertex ordering of G*

with $\mathrm{wcol}_r(G) \leqslant g(r,d)$*, where* $g : \mathbb{N}\times\mathbb{N} \to \mathbb{N}$ *is a fixed computable function.*

Our approach to proving Theorem 4.1 is as follows. The weak coloring numbers are closely related to another measure called admissibility. More precisely, every order witnessing that the r-admissibility is small also witnesses that the weak r-coloring number is small. For computing r-admissibility there exists a greedy approximation algorithm [12, 20]. We turn this greedy approximation algorithm into a low-depth circuit using a similar trick as we did for degeneracy. In each single step of the algorithm we make use of the color coding toolbox provided by Bannach et al. [3].

For applications we will need to efficiently compute the weak r-reachability relation, as expressed in the next lemma.

Lemma 4.2 (⋆). *Consider the following problem parameterized by* r*: Given a graph* G*, its vertex ordering* σ*, and two vertices* u *and* v*, determine whether* u *is weakly* r*-reachable from* v *in* σ*. Then this problem can be solved by a* dlogtime*-uniform family* $(W_{n,r})_{n,r\in\mathbb{N}}$ *of* AC*-circuits where each* $W_{n,r}$ *has size* $n^{O(1)}$ *and depth* $O(\log r)$*.*

Proof. It suffices to verify whether $u \leqslant_\sigma v$ and the distance between u and v in the subgraph induced by vertices not smaller in σ than u is at most r. The former can be read from an input gate, while the latter can be done using a circuit of size $n^{O(1)}$ and depth $O(\log r)$ by Lemma 2.3. □

4.2 Low treedepth colorings

Being able to efficiently compute vertex orderings with low weak coloring numbers enables us to compute low treedepth colorings. Let us now introduce the relevant definitions.

A *separation forest*[2] of a graph G is a forest F on the same vertex set as G such that whenever uv is an edge in G, then either u is an ancestor of v, or v is an ancestor of u in F The *treedepth* of a graph G is the smallest possible depth of a separation forest of G. For an integer p, a coloring $\lambda\colon V(G) \to \{1,\ldots,M\}$ is a *treedepth-*p *coloring* of G if every i-tuple of color classes in λ, $i \leqslant p$, induces in G a graph of treedepth at most i.

It is shown in [23] that a class C of graphs has bounded expansion if and only if for every p there is a number M such that every graph $G \in C$ admits a treedepth-p coloring using M colors. We remark that the above definition of a treedepth-p coloring can be relaxed, yielding a less restrictive definition that is sufficient for most algorithmic purposes, including our purposes in this paper. Namely, it would be sufficient to require that every p-tuple of classes induces in G a graph of treedepth at most $f(p)$, for some function $f : \mathbb{N} \to \mathbb{N}$. It follows from the proof in [23] that this weaker variant yields a notion that is still equivalent to having bounded expansion. Below, we use the original notion of treedepth-p colorings.

We now show how to compute low treedepth colorings orderings with low weak r-coloring numbers.

Suppose G is a graph and σ is a vertex ordering of G. For $r \in \mathbb{N}$, let $G\langle r,\sigma\rangle$ be the *weak* r*-reachability graph* of σ, whose vertex set is $V(G)$ and where $u <_\sigma v$ are considered adjacent if and only if $u \in \mathrm{WReach}_r[G,\sigma,v]$. The following lemma explains the relation between weak r-reachability graphs and low treedepth colorings.

[2]This notion is also called *elimination forest* in the literature; we find the name *separation forest* more explanatory.

Lemma 4.3 (implicit in Theorem 2.6 of [31]). *For any graph* G*, its vertex ordering* σ*, and integer* $p \in \mathbb{N}$*, every proper coloring of* $G\langle 2^{p-2},\sigma\rangle$ *is a treedepth-*p *coloring of* G*.*

Observe that if G is a graph with a vertex ordering σ such that $\mathrm{wcol}_{2^{p-2}}(G,\sigma) \leqslant c$ for some $c \in \mathbb{N}$, then σ is actually a vertex ordering of $G\langle 2^{p-2},\sigma\rangle$ of degeneracy $c-1$. This implies that $G\langle 2^{p-2},\sigma\rangle$ admits a proper coloring with c colors. We already know how to efficiently compute vertex orderings with low weak r-coloring numbers for graphs from any class of bounded expansion, see Theorem 4.1. Applying Lemma 3.6 to the graph $G\langle 2^{p-2},\sigma\rangle$, we get a parallelized algorithm for computing a treedepth-p coloring of a given graph G. We get the following result as an immediate corollary.

Theorem 4.4 (⋆). *Suppose* $\mathscr{C}$ *is a class of effectively bounded expansion. Then for every* $p \in \mathbb{N}$ *there exists a constant* $M = M(p)$*, computable from* p*, such that the following transformation parameterized by* p *is in* para-AC^1*: given a graph* $G \in \mathscr{C}$*, compute its treedepth-*p *coloring using* M *colors.*

Proof. Let $r = 2^{p-2}$. Since $\mathscr{C}$ has effectively bounded expansion, there exists a constant $d \in \mathbb{N}$, computable from p, such that no graph from $\mathscr{C}$ admits a depth-$(r-1)$ topological minor with edge density larger than d. Consequently, given $G \in \mathscr{C}$ we may apply the circuit provided by Theorem 4.1 to compute a vertex ordering σ of G with $\mathrm{wcol}_r(G,\sigma) \leqslant g(r,d)$, where the latter is again a constant computable from p. This circuit has size $f(p,d)\cdot n^{O(1)}$ and depth $O(\log n)$, for computable f. Next, by applying the circuit provided by Lemma 4.2 to each pair of vertices in G we may compute the adjacency matrix of the graph $G\langle r,\sigma\rangle$. Since σ witnesses that $G\langle r,\sigma\rangle$ is $(g(r,d)-1)$-degenerate, by applying the circuit provided by Lemma 3.6 to $G\langle r,\sigma\rangle$ we may compute a proper coloring of this graph using at most $M := (4g(r,d))^2$ colors, which is a constant depending on p in a computable manner. Lemma 4.3 asserts that this coloring is a treedepth-p coloring of G. The claimed size and depth bounds on the obtained circuit follow directly from the construction and from bounds provided by Theorem 4.1, Lemma 4.2, and Lemma 3.6. □

We remark that the problem of computing a low treedepth coloring can be also approached using *fraternal augmentations*, as was done e.g. in [13, 23, 28]. Both in our line of reasoning and in this approach the key step is computing a vertex ordering of low degeneracy, that is, Lemma 3.4. However, we feel that taking this approach would make the argument considerably more technical and would require relying on more involved black-boxes.

4.3 Computing separation forests

A low treedepth coloring is still not enough for the model-checking algorithm to work, as we also need to compute separation forests witnessing that appropriate induced subgraphs have bounded treedepth. In general computing separation forests of optimal depth is a hard computational problem, but if one allows approximate depth there is a very simple and well-known way to do it (see Section 17.3 in [27]), namely, any DFS forest of G provides a separation forest of depth at most 2^h if G has treedepth at most h.

As shown by Bannach et al. [4, Lemma 6], on graphs of bounded treedepth a DFS forest can be computed in para-AC$^{0\uparrow}$ parameterized by treedepth.

Lemma 4.5 (★, see also Lemma 6 of [4]). *The following transformation parameterized by h is in* para-$\mathrm{AC}^{0\uparrow}$*: Given a graph of treedepth h, compute any its DFS forest (represented by its parent relation, encoded as a boolean $n \times n$ matrix).*

5 Model checking

In this section we prove our main result, Theorem 1.1. The general idea of our proof is as follows. We prove the existence of a certain efficient quantifier-elimination procedure for classes of bounded expansion. This is first done in the case of forests of bounded depth. This lifts immediately to classes of bounded treedepth. Finally, this lifts to classes of bounded expansion, via low treedepth colorings.

We assume familiarity with basic notation for relational structures. Unary relations will be also called *labels* for brevity. The *Gaifman graph* of $\mathbb{A}$, denoted $G(\mathbb{A})$, has $V(\mathbb{A})$ as the vertex set, and we make two distinct elements u, v adjacent in $G(\mathbb{A})$ iff u and v appear together in some relation in $\mathbb{A}$.

5.1 Quantifier elimination on forests of bounded depth

In this section we work out a basic primitive for our quantifier elimination procedure, namely the case of unordered, labeled forests of bounded depth.

Rooted forests. Suppose $\mathbb{T}$ is a rooted, unordered forest, so far without any labels. We use standard notions like parent, child, ancestor, descendant, and we follow the convention that each node is regarded as its own ancestor and descendant. The size $|\mathbb{T}|$ of a forest $\mathbb{T}$ is the number of nodes in it. The *depth* of a node x is the number of its ancestors, and the *depth* of a forest $\mathbb{T}$ is the largest depth of a node in $\mathbb{T}$.

Throughout this section we work with forests of depth at most d, for a fixed constant $d \in \mathbb{N}$. A forest $\mathbb{T}$ of depth at most d will be modeled as a relational structure whose universe is the node set and there is one binary relation parent, where $\mathsf{parent}(x, y)$ holds if x is the parent of y.

Proposition 5.1. *There exist existential formulas*

$$\mathsf{lcd}_0(x,y), \mathsf{lcd}_1(x,y), \ldots, \mathsf{lcd}_d(x,y) \in \mathrm{FO}[\mathsf{parent}],$$

each of quantifier depth at most $2d$, such that for every forest $\mathbb{T}$ of depth at most d, $i \in [0,d]$, and nodes x, y we have $\mathbb{T} \models \mathsf{lcd}_i(x,y)$ if and only if x and y have exactly i common ancestors in $\mathbb{T}$.

Observe that the condition in Proposition 5.1 can be equivalently stated as follows: $\mathsf{lcd}_0(x,y)$ holds iff x and y have no common ancestor (i.e. they reside in different trees of the forest), and for $i \geqslant 1$ $\mathsf{lcd}_i(x,y)$ holds iff the least common ancestor of x and y is at depth i in $\mathbb{T}$. Note that for a node x, the formula $\mathsf{lcd}_i(x,x)$ holds if and only if x is at depth i. Moreover, the condition that x is an ancestor of y can be expressed as follows: $\mathsf{lcd}_i(x,y)$ holds iff $\mathsf{lcd}_i(x,x)$ holds, for all $i \in [0,d]$. Thus the formulas lcd_i can be used to check the depths of nodes and the ancestor relation (using boolean combinations).

For a forest $\mathbb{T}$ and a finite label set Λ, a Λ-*labeling* of $\mathbb{T}$ is any structure obtained from $\mathbb{T}$ by adding a unary relation c for each label $c \in \Lambda$. A Λ-labeling of a forest will be also called a Λ-*labeled forest.*

Lcd types. Fix a label set Λ; we consider Λ-labeled forests of depth at most d. A formula $\psi(\bar{x}) \in \mathrm{FO}[\{\mathsf{parent}\} \cup \Lambda]$ is *lcd-reduced* if it uses neither quantifiers nor the parent relation, but it may use formulas lcd_i for $i \in [0,d]$ as atoms. That is, an lcd-reduced formula is a boolean combination of label tests and formulas lcd_i. Note that lcd-reduced formulas are closed under boolean combinations.

We now show how to eliminate a single existential quantifier for bounded depth forests. This will be used later for quantifier elimination on low-treedepth decompositions.

Lemma 5.2 (★). *Let $d \in \mathbb{N}$ and Λ be a label set. Then for every formula $\varphi(\bar{x}) \in \mathrm{FO}[\{\mathsf{parent}\} \cup \Lambda]$ with $|\bar{x}| \geqslant 1$ and of the form*

$$\varphi(\bar{x}) = \exists_y \; \psi(\bar{x}, y)$$

where ψ is lcd-reduced, there exists a label set $\widehat{\Lambda}$, and an lcd-reduced formula $\widehat{\varphi}(\bar{x}) \in \mathrm{FO}[\{\mathsf{parent}\} \cup \widehat{\Lambda}]$, such that for every Λ-labeled forest $\mathbb{T}$ of depth at most d, there is a $\widehat{\Lambda}$-relabeling $\mathbb{S}$ of $\mathbb{T}$ such that $\varphi(\mathbb{T}) = \widehat{\varphi}(\mathbb{S})$.

Moreover, the following effectiveness assertions hold. The label set $\widehat{\Lambda}$ is computable from d and Λ, the formula $\widehat{\varphi}$ is computable from φ, d, Λ, and the following transformation which computes $\mathbb{S}$ given $\mathbb{T}$, parameterized by φ, d, Λ is in linFPT *and in* para-$\mathrm{AC}^{0\uparrow}$.

5.2 Quantifier elimination for bounded expansion classes

Skeletons. We first introduce *skeletons*, which are relational structures formed by putting a bounded number of forests of bounded depth on top of each other. Essentially, they will be our abstraction for low treedepth decompositions.

Let Γ be a vocabulary (of arity 2) and let $d \in \mathbb{N}$. A Γ-structure $\mathbb{A}$ is a Γ-*skeleton of depth d* if for every binary relation $R \in \Gamma$, the structure obtained from $\mathbb{A}$ by dropping all relations apart from R and preserving the universe is a rooted forest of depth at most d, with R serving the role of the parent relation. Note that thus *all* binary relations in Γ serve the roles of bounded-depth forests.

For every binary relation $R \in \Gamma$ and $i \in [0,d]$ we may construct a formula $\mathsf{lcd}_i^R(x,y)$ as in Proposition 5.1, but using R instead of parent. As before, a formula $\varphi(\bar{x}) \in \mathrm{FO}[\Gamma]$ is *lcd-reduced* if it does not use any quantifiers or binary relations, but may use formulas lcd_i^R as atoms; thus, it is a boolean combination of label checks and atoms lcd_i^R. We note that lcd-reduced formulas can be easily turned into existential formulas. For future reference we observe that lcd-reduced formulas can be efficiently evaluated.

Lemma 5.3 (★). *The following problem parameterized by $d \in \mathbb{N}$ and an lcd-reduced formula $\alpha(\bar{x}) \in \mathrm{FO}[\Gamma]$ is in* para-$\mathrm{AC}^{0\uparrow}$ *and can be computed in time $O(d|\alpha|)$: given a Γ-skeleton $\mathbb{A}$ of depth d and a tuple $\bar{u} \in V(\mathbb{A})^{|\bar{x}|}$, verify whether $\mathbb{A} \models \alpha(\bar{u})$.*

Guarded structures. For the remainder of this section we fix a graph class $\mathscr{C}$ with effectively bounded expansion. Without loss of generality we may assume that $\mathscr{C}$ is closed under taking subgraphs. Let us fix the function $M(\cdot)$ given by Theorem 4.4 for the class $\mathscr{C}$; this means that given a graph $G \in \mathscr{C}$ and parameter p we may compute a treedepth-p coloring of G using at most $M(p)$ colors in para-AC^1.

A structure $\mathbb{A}$ is *guarded* by $\mathscr{C}$ if the Gaifman graph of $\mathbb{A}$ belongs to $\mathscr{C}$. Further, a structure $\mathbb{B}$ with the same universe as $\mathbb{A}$ (but possibly different vocabulary) is *guarded* by $\mathbb{A}$ if the Gaifman graph of $\mathbb{B}$ is a subgraph of the Gaifman graph of $\mathbb{A}$. Note that if $\mathbb{A}$ guards $\mathbb{B}$ and $\mathbb{B}$ guards $\mathbb{C}$ then $\mathbb{A}$ guards $\mathbb{C}$, and if further $\mathbb{A}$ is guarded by $\mathscr{C}$, then so are $\mathbb{B}$ and $\mathbb{C}$.

Quantifier elimination. We finally proceed to our main goal, the quantifier elimination procedure for FO on structures with Gaifman graphs from $\mathscr{C}$. The following definition explains our goal in this procedure.

Definition 5.4. Let Σ be a vocabulary (of arity 2) and let $\varphi(\bar{x}) \in \mathrm{FO}[\Sigma]$ for a tuple of variables $\bar{x} = (x_1, \ldots, x_k)$. We say that $\varphi(\bar{x})$ is *reducible* if there exists $d \in \mathbb{N}$, a vocabulary Γ, an lcd-reduced formula $\alpha(\bar{x}) \in \mathrm{FO}[\Gamma]$, and, for every Σ-structure $\mathbb{A}$ guarded by $\mathscr{C}$, a Γ-skeleton $\mathbb{B}$ of depth at most d guarded by $\mathbb{A}$ such that $\varphi(\mathbb{A}) = \alpha(\mathbb{B})$.

This reducibility is *effective* if d, Γ, and α are computable from Σ and φ, and the transformation computing $\mathbb{B}$ given $\mathbb{A}$, parameterized by Σ and φ, is in linFPT and para-AC^1.

Note that in Definition 5.4, the fact that $\mathbb{A}$ is guarded by $\mathscr{C}$ and guards $\mathbb{B}$, entails that $\mathbb{B}$ is guarded by $\mathscr{C}$. Hence we may further apply further reducibility on the structure $\mathbb{B}$ and so on. This chaining property of the notion of reducibility will be crucial in our reasoning.

In the following, whenever a vocabulary of a formula is not specified, it is an arbitrary vocabulary. Given Definition 5.4, quantifier elimination can be stated in a very simple way.

Theorem 5.5. *Every formula $\varphi(\bar{x})$ with $|\bar{x}| \geqslant 1$ is effectively reducible.*

The proof of Theorem 5.5 is by induction on the structure of the formula. We find it most convenient to directly solve the case of existential formulas first, from which both the induction base and the induction step will follow.

Lemma 5.6. *Every existential formula with at least one free variable is effectively reducible.*

Proof. Let $\varphi(\bar{x}) \in \mathrm{FO}[\Sigma]$ be the formula in question, where Σ is a vocabulary and $\bar{x} = (x_1, \ldots, x_k)$ are the free variables of φ. We may assume that φ is in prenex existential form, say, $\varphi(\bar{x}) = \exists_{\bar{y}}\, \psi(\bar{x}, \bar{y})$, where $\bar{y} = (y_1, \ldots, y_\ell)$ and $\psi(\bar{x}, \bar{y})$ is quantifier-free. Suppose $\mathbb{A}$ is the given Σ-structure guarded by $\mathscr{C}$. We describe how a suitable skeleton $\mathbb{B}$ guarded by $\mathbb{A}$ should be constructed, while its depth d, its vocabulary Γ, and the final lcd-reduced formula $\alpha(\bar{x})$ will be constructed along the way.

Let $p = k + \ell$ and let $G = G(\mathbb{A})$. Since $G \in \mathscr{C}$, there is a treedepth-p coloring $\lambda \colon V(\mathbb{A}) \to [M]$ of G, where $M = M(p)$, which can be computed in para-AC^1 and in linFPT using Theorem 4.4 and the results of [23, 26], respectively. Let $\mathcal{U}$ be the family of all subsets of $[M]$ of size p. For $C \in \mathcal{U}$, let $V^C = \lambda^{-1}(C)$ be the set of elements with colors from C, and let $G^C = G[V^C]$ be the subgraph induced by them. Then G^C has treedepth at most p. As observed before, we may compute, for each $C \in \mathcal{U}$, a DFS forest F^C of the induced subgraph G^C of depth at most $d := 2^p - 1$ in para-AC$^{0\uparrow}$ (and in linFPT by just running a depth-first search).

Fix any $C \in \mathcal{U}$ and let $\mathbb{T}^C$ be the unlabeled forest of depth at most d with node set V^C and binary relation parent^C interpreted as the parent relation of F^C. We now prove that the substructure induced in $\mathbb{A}$ by V^C can be entirely encoded in a labeling of $\mathbb{T}^C$.

Claim 1 ($\star$). *There exists a label set Λ^C, a Λ^C-labeling $\mathbb{S}^C$ of $\mathbb{T}^C$, and, for every relation $R \in \Sigma$, an lcd-reduced formula η_R^C with as many free variables as the arity of R such that $R(\mathbb{A}[V^C]) = \eta_R^C(\mathbb{S}^C)$, where $\mathbb{A}[V^C]$ denotes the substructure of $\mathbb{A}$ induced by V^C.*

Now consider formula $\psi^C(\bar{x}, \bar{y}) \in \mathrm{FO}[\{\mathrm{parent}^C\} \cup \Lambda^C]$ obtained from $\psi(\bar{x}, \bar{y})$ by replacing each relation symbol R with the corresponding formula η_R^C. Since ψ was quantifier-free, ψ^C is lcd-reduced. Let

$$\varphi^C(\bar{x}) := \exists_{\bar{y}}\, \psi^C(\bar{x}, \bar{y}).$$

Since $k \geqslant 1$, we may iteratively apply Lemma 5.2 to consecutive quantifiers in φ^C, starting with the deepest. This yields a new label set $\widehat{\Lambda}^C$, a $\widehat{\Lambda}^C$-relabeling $\mathbb{U}^C$ of $\mathbb{S}^C$, and an lcd-reduced formula $\alpha^C(\bar{x})$ such that $\alpha^C(\mathbb{U}^C) = \varphi^C(\mathbb{S}^C)$.

We now build $\mathbb{B}$ and its vocabulary Γ. Start by setting the universe of $\mathbb{B}$ to be equal to the universe of $\mathbb{A}$, and Γ is so far empty. For each $C \in \mathcal{U}$ add a unary relation class^C to Γ, and interpret it in $\mathbb{B}$ so that it selects the vertices of V^C. Next, import all the relations from all structures $\mathbb{U}^C$ to $\mathbb{B}$. That is, for each $C \in \mathcal{U}$ we add $\{\mathrm{parent}^C\} \cup \widehat{\Lambda}^C$ to the vocabulary Γ, while the interpretations of these relations are taken from $\mathbb{S}^C$. Note that thus elements outside of V^C do not participate in relations parent^C.

This concludes the construction of Γ and $\mathbb{B}$. Note that the only binary relations in Γ are the relations parent^C for $C \in \mathcal{U}$, and in $\mathbb{B}$ each of them induces a forest of depth at most d. Moreover, since each F^C is a DFS forest of G^C, it follows that $\mathbb{B}$ is guarded by $\mathbb{A}$. Hence, $\mathbb{B}$ is a Γ-skeleton of depth d guarded by $\mathbb{A}$, as requested. Consider the formula

$$\alpha(\bar{x}) := \bigvee_{C \in \mathcal{U}} \left(\alpha^C(\bar{x}) \wedge \bigwedge_{i=1}^{k} \mathrm{class}^C(x_i) \right).$$

Observe that $\alpha(\bar{x})$, as a boolean combination of lcd-reduced formulas, is lcd-reduced. We claim that $\alpha(\mathbb{B}) = \varphi(\mathbb{A})$. On one hand, by the construction it is clear that α selects only tuples that satisfy φ, thus $\alpha(\mathbb{B}) \subseteq \varphi(\mathbb{A})$. To see the reverse inclusion, observe that whenever we have some valuation $\bar{u}$ of $\bar{x}$ such that $\varphi(\bar{u})$ holds, this is witnessed by the existence of some valuation $\bar{v}$ of $\bar{y}$ such that $\psi(\bar{u}, \bar{v})$ holds. Since $|\bar{u}| + |\bar{v}| = k + \ell = p$ and λ was a treedepth-p coloring, there exists $C \in \mathcal{U}$ such that all elements of $\bar{u}$ and $\bar{v}$ belong to V^C. For this C the formula $\alpha^C(\bar{u}) \wedge \bigwedge_{i=1}^{k} \mathrm{class}^C(u_i)$ will be satisfied and, consequently, $\bar{u}$ will be included in $\alpha(\mathbb{B})$.

This proves reducibility. Effectiveness follows immediately from complexity bounds provided by the invoked results and a straightforward implementation of the construction of Claim 1. ◻

We now use Lemma 5.6 to give all the ingredients needed for the inductive proof of Theorem 5.5.

Lemma 5.7 ($\star$). *The following assertions hold:*

(a) *Every quantifier-free formula with at least one free variable is effectively reducible.*
(b) *The negation of an effectively reducible formula is effectively reducible.*
(c) *Every formula of the form $\varphi(\bar{x}) = \exists_y\, \psi(\bar{x}, y)$ for an effectively reducible ψ and with $|\bar{x}| \geqslant 1$ is also effectively reducible.*

5.3 Piecing together the proof of Theorem 1.1

With quantifier elimination in place, we may conclude the proof of our main result, Theorem 1.1.

Proof of Theorem 1.1. Let $\mathbb{A}$ be an input Σ-structure on n elements, for a vocabulary Σ of arity at most 2, and let $\varphi \in \mathrm{FO}[\Sigma]$ be the input sentence. We would like to apply Theorem 5.5 to φ. However there is a slight mismatch: Theorem 5.5 assumes that the input formula has at least one free variable. To circumvent this, let z be a fresh variable that is not used in φ and let us consider φ as a formula $\varphi(z)$ with one free variable z that is never used. Note that $\varphi(z)$ is true either for every element of $\mathbb{A}$ or for no element of $\mathbb{A}$, depending on whether φ is true or false in $\mathbb{A}$. Now apply Theorem 5.5 to $\varphi(z)$, yielding a Γ-skeleton $\mathbb{B}$ of some depth d guarded by $\mathbb{A}$ and an lcd-reduced formula $\alpha(z) \in \mathrm{FO}[\Gamma]$ such that $\alpha(\mathbb{B}) = \varphi(\mathbb{A})$. It remains to evaluate α on any element of the structure using Lemma 5.3. □

The same reasoning allows to reprove the result of Dvořák et al. [13] that for every class of effectively bounded expansion $\mathscr{C}$, it can be verified in linear-FPT time whether an input sentence φ holds in a given structure whose Gaifman graph belongs to $\mathscr{C}$.

6 Conclusions

In this paper we showed that the model-checking problem for first-order logic on classes of effectively bounded expansion is in para-AC^1, which means that it can be solved by a family of AC-circuits of size $f(\varphi) \cdot n^{O(1)}$ and depth $f(\varphi) + O(\log n)$, where f is a computable function. This can be regarded as a parallelized variant of the result of Dvořák et al. [13] stating that the problem is fixed-parameter tractable.

By the result of Grohe et al. [21], model-checking FO is fixed-parameter tractable even on every nowhere dense class of structures. When trying to generalize our result to the nowhere dense setting, the main issue is that the proof of Grohe et al. [21] does not yield a robust quantifier elimination procedure, but a weak variant of Gaifman local form that is sufficient for fixed-parameter tractability of model-checking, but not variations of the problem.

Our techniques uncover tight connections between the paradigms of distributed computing and circuit complexity in the context of sparse graphs classes. Methods of the theory of sparsity seem very well-suited for the design of distributed algorithms, yet so far little is known. Nešetřil and Ossona de Mendez gave a logarithmic-time distributed algorithm to compute low treedepth colorings on classes of bounded expansion [28]. In the light of this paper, it is very natural to repeat the question asked by Nešetřil and Ossona de Mendez [28] of whether on every class of bounded expansion, model-checking local first-order formulas can be performed by a distributed algorithm with running time $f(\varphi) \cdot \log n$ in the *local broadcast* model. As computation of low treedepth colorings is already settled [28], it remains to examine the quantifier elimination procedure; we hope that our presentation of this argument may help with this. Stronger models of communication (such as the so-called *congested clique* model) may allow efficient distributed algorithms for more general problems, like model-checking of first-order formulas that are not necessarily local.

Acknowledgements. The authors thank Thomas Zeume for discussions on dynamic FO in the context of sparse graphs, which inspired this work.

References

[1] Srinivasa Rao Arikati, Anil Maheshwari, and Christos D. Zaroliagis. 1997. Efficient Computation of Implicit Representations of Sparse Graphs. *Discrete Applied Mathematics* 78, 1-3 (1997), 1–16.

[2] Baruch Awerbuch, Andrew V. Goldberg, Michael Luby, and Serge A. Plotkin. 1989. Network Decomposition and Locality in Distributed Computation. In *FOCS 1989*. IEEE Computer Society, 364–369.

[3] Max Bannach, Christoph Stockhusen, and Till Tantau. 2015. Fast Parallel Fixed-parameter Algorithms via Color Coding. In *IPEC 2015 (LIPIcs)*, Vol. 43. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 224–235.

[4] Max Bannach and Till Tantau. 2016. Parallel Multivariate Meta-Theorems. In *IPEC 2016 (LIPIcs)*, Vol. 63. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 4:1–4:17.

[5] Max Bannach and Till Tantau. 2017. Computing Hitting Set Kernels By AC^0-Circuits. (2017). Manuscript, accepted for publication at STACS 2018.

[6] Leonid Barenboim. 2016. Deterministic $(\Delta + 1)$-Coloring in Sublinear (in Δ) Time in Static, Dynamic, and Faulty Networks. *J. ACM* 63, 5 (2016), 47:1–47:22.

[7] Leonid Barenboim and Michael Elkin. 2010. Sublogarithmic distributed MIS algorithm for sparse graphs using Nash-Williams decomposition. *Distributed Computing* 22, 5-6 (2010), 363–379.

[8] Yijia Chen, Jörg Flum, and Xuangui Huang. 2017. Slicewise Definability in First-Order Logic with Bounded Quantifier Rank. In *CSL 2017 (LIPIcs)*, Vol. 82. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 19:1–19:16.

[9] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2015. *Parameterized Algorithms*. Springer.

[10] Anuj Dawar, Martin Grohe, and Stephan Kreutzer. 2007. Locally Excluding a Minor. In *LICS 2007*. IEEE Computer Society, 270–279.

[11] Rodney G. Downey and Michael R. Fellows. 2013. *Fundamentals of Parameterized Complexity*. Springer.

[12] Zdeněk Dvořák. 2013. Constant-factor approximation of the domination number in sparse graphs. *European Journal of Combinatorics* 34, 5 (2013), 833–840.

[13] Zdeněk Dvořák, Daniel Král', and Robin Thomas. 2013. Testing first-order properties for subclasses of sparse graphs. *Journal of the ACM (JACM)* 60, 5 (2013), 36.

[14] Michael Elberfeld, Christoph Stockhusen, and Till Tantau. 2015. On the Space and Circuit Complexity of Parameterized Problems: Classes and Completeness. *Algorithmica* 71, 3 (2015), 661–701.

[15] Jörg Flum and Martin Grohe. 2001. Fixed-Parameter Tractability, Definability, and Model-Checking. *SIAM J. Comput.* 31, 1 (2001), 113–145.

[16] Jörg Flum and Martin Grohe. 2006. *Parameterized Complexity Theory*. Springer.

[17] Markus Frick and Martin Grohe. 2001. Deciding first-order properties of locally tree-decomposable structures. *J. ACM* 48, 6 (2001), 1184–1206.

[18] Andrew V. Goldberg, Serge A. Plotkin, and Gregory E. Shannon. 1987. Parallel Symmetry-Breaking in Sparse Graphs. In *STOC 1987*. ACM, 315–324.

[19] Martin Grohe and Stephan Kreutzer. 2011. Methods for Algorithmic Meta Theorems. In *Model Theoretic Methods in Finite Combinatorics*, M. Grohe and J.A. Makowsky (Eds.). Contemporary Mathematics, Vol. 558. American Mathematical Society, 181–206.

[20] Martin Grohe, Stephan Kreutzer, Roman Rabinovich, Sebastian Siebertz, and Konstantinos Stavropoulos. 2015. Colouring and covering nowhere dense graphs. In *WG 2015*. Springer, 325–338.

[21] Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. 2017. Deciding First-Order Properties of Nowhere Dense Graphs. *Journal of the ACM (JACM)* 64, 3 (2017), 17:1–17:32.

[22] Neil Immerman. 1999. *Descriptive complexity*. Springer.

[23] Jaroslav Nešetřil and Patrice Ossona de Mendez. 2008. Grad and classes with bounded expansion I. Decompositions. *European Journal of Combinatorics* 29, 3 (2008), 760–776.

[24] Jaroslav Nešetřil and Patrice Ossona de Mendez. 2010. First order properties on nowhere dense structures. *The Journal of Symbolic Logic* 75, 03 (2010), 868–887.

[25] Jaroslav Nešetřil and Patrice Ossona de Mendez. 2011. On nowhere dense graphs. *European Journal of Combinatorics* 32, 4 (2011), 600–617.

[26] Jaroslav Nešetřil and Patrice Ossona de Mendez. 2008. Grad and classes with bounded expansion II. Algorithmic aspects. *European Journal of Combinatorics* 29, 3 (2008), 777–791.

[27] Jaroslav Nešetřil and Patrice Ossona de Mendez. 2012. *Sparsity — Graphs, Structures, and Algorithms*. Algorithms and Combinatorics, Vol. 28. Springer.

[28] Jaroslav Nešetřil and Patrice Ossona de Mendez. 2016. A distributed low tree-depth decomposition algorithm for bounded expansion classes. *Distributed Computing* 29, 1 (2016), 39–49.

[29] Michał Pilipczuk and Sebastian Siebertz. Winter Semester 2017/18. Lecture notes for the course "Sparsity" given at Faculty of Mathematics, Informatics, and Mechanics of the University of Warsaw. (Winter Semester 2017/18). Available at https://www.mimuw.edu.pl/~mp248287/sparsity.

[30] Detlef Seese. 1995. Linear time computable problems and logical descriptions. *Electr. Notes Theor. Comput. Sci.* 2 (1995), 246–259.

[31] Xuding Zhu. 2009. Colouring graphs with bounded generalized colouring number. *Discrete Mathematics* 309, 18 (2009), 5562–5568.